



UNIVERSIDADE FEDERAL DO CEARÁ
CENTRO DE TECNOLOGIA
DEPARTAMENTO DE ENGENHARIA DE TELEINFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA DE TELEINFORMÁTICA
DOUTORADO EM ENGENHARIA DE TELEINFORMÁTICA

DAVID FREITAS MOURA MOTA

**ROBUST OBC: UM SISTEMA DE COMPUTAÇÃO DE BORDO TOLERANTE A
FALHAS**

FORTALEZA

2022

DAVID FREITAS MOURA MOTA

ROBUST OBC: UM SISTEMA DE COMPUTAÇÃO DE BORDO TOLERANTE A FALHAS

Tese apresentada ao Programa de Pós-Graduação em Engenharia de Teleinformática do Centro de Tecnologia da Universidade Federal do Ceará, como requisito parcial à obtenção do título de doutor em Engenharia de Teleinformática. Área de Concentração: Sinais e Sistemas

Orientador: Prof. Dr. João Cesar Moura Mota

Coorientador: Prof. Dr. Fabian Luis Vargas

FORTALEZA

2022

Dados Internacionais de Catalogação na Publicação
Universidade Federal do Ceará
Biblioteca Universitária

Gerada automaticamente pelo módulo Catalog, mediante os dados fornecidos pelo(a) autor(a)

M871r Mota, David Freitas Moura.
ROBUST OBC: UM SISTEMA DE COMPUTAÇÃO DE BORDO TOLERANTE A FALHAS / David
Freitas Moura Mota. – 2022.
143 f. : il. color.

Tese (doutorado) – Universidade Federal do Ceará, Centro de Tecnologia, Programa de Pós-Graduação
em Engenharia de Teleinformática, Fortaleza, 2022.

Orientação: Prof. Dr. João Cesar Moura Mota.

Coorientação: Prof. Dr. Fabian Luis Vargas.

1. CubeSat. 2. Computador de bordo. 3. Rad-Hard. 4. COTS. I. Título.

CDD 621.38

DAVID FREITAS MOURA MOTA

ROBUST OBC: UM SISTEMA DE COMPUTAÇÃO DE BORDO TOLERANTE A FALHAS

Tese apresentada ao Programa de Pós-Graduação em Engenharia de Teleinformática do Centro de Tecnologia da Universidade Federal do Ceará, como requisito parcial à obtenção do título de doutor em Engenharia de Teleinformática. Área de Concentração: Sinais e Sistemas

Aprovada em: 22 de Fevereiro de 2022

BANCA EXAMINADORA

Prof. Dr. João Cesar Moura Mota (Orientador)
Universidade Federal do Ceará (UFC)

Prof. Dr. Fabian Luis Vargas (Coorientador)
Pontifícia Universidade Católica do Rio Grande
do Sul (PUC-RS)

Profa. Dra. Lirida Alves de Barros Naviner
Telecom Paris, França

Prof. Dr. Raul de Lacerda
Centrale-Supelec, França

Profa. Dra. Maria De Fatima Mattiello
Instituto Nacional de Pesquisas Espaciais
(INPE)

Dedico essa tese a todos aqueles que de alguma forma contribuíram nesse momento tão difícil e cheio de renúncias. Pessoas que me ajudaram diretamente no trabalho e pessoas que ajudaram indiretamente me dando forças e me fazendo sorrir mesmo quando estava prestes a desistir. Muito obrigado.

AGRADECIMENTOS

Pai, mais uma vez, muito obrigado pela força, dedicação e, principalmente, por acreditar em mim, me motivando e me ajudando de todas as maneiras possíveis para que esse objetivo fosse alcançado.

Minha querida mãe, eu sei que a senhora já passou por essa situação antes e foi muito sacrificante, porém lhe agradeço por tudo e acredito que isso tem um propósito. Sua garra é o maior exemplo que eu poderia ter.

Minha linda, obrigado por estar comigo, obrigado por abdicar de tudo pela nossa família, obrigado por fazer dos meus dias mais felizes, obrigado por me dar meus maiores presentes que são nossos filhos. Obrigado por tudo, eu te amo.

Meus filhos, todos esses esforços são por e para vocês e acredito que em um futuro próximo irão entender. Vocês me dão forças para eu seguir em frente, sem demonstrar cansaço, para que a gente possa ser a cada dia uma família mais feliz.

Meus irmãos, sobrinhos, tios e primos, muito obrigado por serem minha família, mesmo nesse momento tão difícil de distanciamento social eu sei que nossos laços jamais irão se romper.

Ao Prof. Dr. Alexandre pelo incentivo e conselhos ao longo do doutorado, muito obrigado.

Aos amigos Júlio, Caio e David por me ajudarem no desenvolvimento desse projeto e pela suas amizades, algo tão valioso e extinto nos dias de hoje. Serei eternamente grato.

Ao Prof. Dr. Fabian pela orientação e ajustes nesse trabalho, muito obrigado.

Aos professores da banca Dr. Helano de Castro, Dr. Paulo Cortez, Dr. Giovanni Barroso e ao Dr. Antonio Macílio pelas contribuições de grande importância nessa tese.

Ao Instituto Nacional de Pesquisas Espaciais (INPE) pelo financiamento do projeto que deu origem a essa tese.

Ao Instituto de Educação Física e Esportes da Universidade Federal do Ceará por acreditar em mim e me liberar meio período do expediente de trabalho para que eu me dedicasse a obtenção do título de doutor.

“Você pode encarar um erro como uma besteira a ser esquecida ou como um resultado que aponta para uma nova direção.”

(Steve Jobs)

RESUMO

O uso de nano satélites que utilizam o padrão CubeSat vem crescendo atualmente devido ao seu baixo custo de fabricação e lançamento em relação aos satélites tradicionais. Além disso, este tipo de nano satélites possui uma grande variedade de aplicações, que vão desde aplicações simples até críticas. Para aplicações críticas, é necessário que os subsistemas do nano satélite tenham algum tipo de proteção contra a radiação e seus efeitos (Rad-Hard), pois falhas severas podem inviabilizar a missão do mesmo. Principalmente o computador de bordo (OBC), o qual é considerado o cérebro do nano satélite e necessita de uma atenção especial no que diz respeito a técnicas de tolerância a falhas. O uso de componentes Rad-Hard é a solução de maior confiabilidade para se propor em projetos de um OBC para aplicações críticas, porém o alto custo de aquisição e o difícil acesso a esses componentes impossibilitam, por parte das universidades e pequenas empresas, a pesquisa e desenvolvimento de OBCs que utilizam este tipo de componente. Com isso, essa tese propõe uma arquitetura de um OBC tolerante a falhas que faz uso de somente componentes COTS, no qual o uso de técnicas de tolerância a falhas propostas resultam em um incremento na confiabilidade, de forma a, a exemplo de componentes Rad-Hard, proporcionar a OBCs um alto grau de tolerância a falhas. Para isso, foram pesquisadas e implementadas técnicas de tolerância a falhas que possam trazer confiabilidade ao OBC como, por exemplo, um sistema de chaveamento de processadores, um gerenciador de falhas que pudesse verificar se as falhas são permanentes ou transitórias e, assim, realizar uma tomada de decisão, assim como outros sistemas como sistema de proteção de memória SRAM e sistema que gerencia o chaveamento de redundância de OBCs. Os testes experimentais realizados nos sistemas de tolerância a falhas do OBC proposto obtiveram resultados que atenderam aos requisitos descritos nesta tese, que se baseiam em três fundamentos principais: baixo consumo de potência elétrica, alto desempenho computacional e alta confiabilidade. Adicionalmente, sua confiabilidade mostrou-se compatível com os OBCs existentes tanto da comunidade acadêmica como do meio industrial. Neste sentido, essa tese descreve uma arquitetura de um OBC de alta confiabilidade, que possui como diferencial técnicas de tolerância a falhas capaz de prover continuidade de funcionamento do subsistema de computação de bordo mesmo na presença de falhas. Em vistas dos resultados pode-se concluir que os objetivos foram alcançados pois, além de proposto um OBC tolerante a falhas baseado em COTS, ainda foram atingidos objetivos secundários como baixo consumo de energia, alto desempenho e reconfigurabilidade do sistema.

Palavras-chave: CubeSat. Computador de bordo. Rad-Hard. COTS.

ABSTRACT

The use of nanosatellites that utilize the CubeSat standard is currently growing due to their low manufacturing and launch costs compared to traditional satellites. In addition, these types of nanosatellites have a wide variety of applications, ranging from simple to critical applications. For critical applications, it can be necessary that the nanosatellite subsystems have some protection against radiation and its effects (Rad-Hard) because severe failures can make its mission almost impossible to accomplish. Especially the onboard computer (OBC), which is considered the brain of the nanosatellite and needs special attention regarding fault tolerance techniques. Rad-Hard components are the most reliable solution to propose in designing an OBC for critical applications. However, the high cost and limited access to these components make it almost impossible for universities and small companies to research and develop OBCs that use this type of component. With this in mind, this thesis proposes an architecture for a fault-tolerant OBC that uses only COTS components, where the use of the proposed fault tolerance techniques results in an increase in reliability, so as to increase the degree of fault tolerance of an OBC, similar to what occurs when using Rad-Hard components. For this purpose, fault tolerance techniques that could bring reliability to the onboard computer were researched and implemented, such as a processor redundancy switching system, a fault manager that could verify if the faults are permanent or transient, as well as other systems such as SRAM memory protection system and a system that manages the redundancy switching of OBCs. The experimental tests performed on the proposed OBC fault tolerance systems obtained results that addressed the requirements described in this thesis, which were based on three main fundamentals: low electrical power consumption, high computational performance, and high reliability. Additionally, its reliability is compatible with existing OBCs from academic and industrial communities. In this sense, this thesis describes an architecture of a high-reliability OBC, which has differential fault tolerance techniques capable of providing continuity of operation of the onboard computing subsystem even in the presence of failures. In the face of the results, it can be concluded that the objectives of this thesis were achieved because not only a fault-tolerant OBC based on COTS was proposed, but also secondary objectives such as low power consumption, high performance, and reconfigurability of the system were achieved.

Keywords: CubeSat, Onboard computer, Rad-Hard, COTS.

LISTA DE FIGURAS

Figura 1 – Formação de unidades do padrão CubeSat	21
Figura 2 – Curva de taxa de falhas em projetos de <i>hardware</i> (<i>Bath Tube</i>).	31
Figura 3 – Modelo de um sistema em Série.	32
Figura 4 – Modelo de um sistema em Paralelo.	33
Figura 5 – Modelo de Múltipla Redundância.	34
Figura 6 – Diagrama de transição de estados da Cadeia de Markov.	35
Figura 7 – Exemplo de um Diagrama de Árvore.	37
Figura 8 – Processo de Interleaving.	42
Figura 9 – Arquitetura Centralizada.	45
Figura 10 – Arquitetura Distribuída.	46
Figura 11 – Arquitetura com Barramento Compartilhado.	46
Figura 12 – Fluxograma de seleção dos OBCs.	48
Figura 13 – Arquitetura do OBC em Busch e Schilling (2013).	49
Figura 14 – Arquitetura do OBC em Leppinen <i>et al.</i> (2014).	51
Figura 15 – Arquitetura do OBC em Botma (2011).	52
Figura 16 – Arquitetura do OBC em Cube Space (2016).	53
Figura 17 – Arquitetura do OBC em Tian <i>et al.</i> (2012).	54
Figura 18 – Arquitetura do OBC em Fayyaz <i>et al.</i> (2012).	56
Figura 19 – Esquema de conexão do AMFT em Fayyaz <i>et al.</i> (2012).	56
Figura 20 – Arquitetura do OBC em Lim (2009).	57
Figura 21 – Arquitetura do OBC em Gauss Srl (2017).	59
Figura 22 – Arquitetura do OBC em Data Patterns (2021).	60
Figura 23 – Modelagem da arquitetura do OBC em Busch e Schilling (2013).	61
Figura 24 – Modelo de Markov para o OBC em Busch e Schilling (2013).	62
Figura 25 – Modelagem da arquitetura do OBC em Leppinen <i>et al.</i> (2014).	62
Figura 26 – Modelagem da arquitetura do OBC em Botma <i>et al.</i> (2013).	64
Figura 27 – Modelo de Markov para o OBC em Botma <i>et al.</i> (2013).	64
Figura 28 – Modelagem da arquitetura do OBC em Tian <i>et al.</i> (2012).	65
Figura 29 – Modelo de Markov para o OBC em Tian <i>et al.</i> (2012).	66
Figura 30 – Modelagem da arquitetura do OBC em Fayyaz <i>et al.</i> (2012).	67
Figura 31 – Modelagem da arquitetura do OBC em Gauss Srl (2017).	68

Figura 32 – Modelagem da arquitetura do OBC em Data Patterns (2021).	68
Figura 33 – Arquitetura de <i>hardware</i> do Robust OBC.	73
Figura 34 – Arquitetura do Sistema de troca de processador do Robust OBC.	77
Figura 35 – Arquitetura do módulo Gerenciador de <i>watchdog</i> do Robust OBC.	78
Figura 36 – Arquitetura do Gerenciador de falhas do Robust OBC.	80
Figura 37 – Arquitetura do Sistema de troca de placa do Robust OBC.	82
Figura 38 – Arquitetura do Sistema de alimentação inteligente do Robust OBC.	83
Figura 39 – Arquitetura para falhas permanentes e funcionais das memórias SRAM.	85
Figura 40 – Modelagem da arquitetura do Robust OBC.	86
Figura 41 – Modelagem do Sistema de troca de placa do Robust OBC.	88
Figura 42 – Esquema de transição de falhas do Robust OBC.	91
Figura 43 – Diagrama de falhas do Robust OBC.	93
Figura 44 – Placa de circuito impresso do Robust OBC com medidas de 9 x 9 cm.	95
Figura 45 – <i>Software</i> de simulação do Sistema de troca de processador.	96
Figura 46 – Diagrama de tempo dos sinais do Sistema de troca de processador.	98
Figura 47 – Máquina de estados do Sistema de troca de processador.	99
Figura 48 – Sinais de entrada e saída do Sistema de troca de processador.	99
Figura 49 – <i>Setup</i> de teste para validação do Sistema de troca de processador.	101
Figura 50 – Máquina de estado do funcionamento do BSS no processador externo.	103
Figura 51 – Processo de troca de placa do Sistema de troca de placa.	104
Figura 52 – Máquina de estado do funcionamento do BSS no processador interno.	105
Figura 53 – <i>Setup</i> de teste do Sistema de troca de placa.	106
Figura 54 – Gráfico de comparação das confiabilidades para falhas permanentes dos OBCs.	110
Figura 55 – Gráfico de comparação das confiabilidades para falhas transitórias dos OBCs.	112
Figura 56 – Gráfico de comparação das confiabilidades para falhas permanentes na redundância de OBCs.	113

LISTA DE TABELAS

Tabela 1 – Tabela com os OBCs selecionados	69
--	----

LISTA DE ABREVIATURAS E SIGLAS

COTS	<i>Commercial Off-The-Shelf</i>
GCR	<i>Galactic Cosmic Radiation</i>
SEE	<i>Single Event Effects</i>
TID	<i>Total Ionizing Dose</i>
SEU	<i>Single Event Upset</i>
SEL	<i>Single Event Latchup</i>
Rad-hard	<i>Radiation Hardening</i>
I2C	<i>Inter-Integrated Circuit</i>
CAN	<i>Controller Area Network</i>
UART	<i>Universal Asynchronous Receiver/Transmitter</i>
OBC	<i>On Board Computer</i>
CDHS	<i>Command and Data Handling Subsystem</i>
FPGA	<i>Field-Programmable Gate Array</i>
SRAM	<i>Static Random Access Memory</i>
SET	<i>Single Event Transient</i>
SEFI	<i>Single Event Functional Interrupt</i>
EDC	<i>Error Detecting Codes</i>
EDAC	<i>Error Detection And Correction</i>
PCI	Placa de Circuito Impresso
SPF	<i>Single Point Failure</i>
OTP	<i>One-Time Programmable</i>
LEO	<i>Low Earth Orbit</i>
MTTF	<i>Mean Time To Failure</i>
MTBF	<i>Mean Time Between Failure</i>
MTTR	<i>Mean Time to Repair</i>
FMEA	<i>Failure Mode and Effect Analysis</i>
CMOS	<i>Complementary Metal-Oxide-Semiconductor</i>
SBU	<i>Single Bit Upset</i>
MBU	<i>Multiple Bit Upset</i>
LESC	Laboratório de Engenharia de Sistemas de Computação

CLC	<i>Column Line Code</i>
PCoSA	<i>Product Code for Space Applications</i>
DAEC	<i>Double Adjacent Error Correction</i>
TAEC	<i>Triple Adjacent Error Correction</i>
QUAEC	<i>Quadruple Adjacent Error Correction</i>
LBC	<i>Linear Block Code</i>
EBI	<i>External Bus Interface</i>
TMR	<i>Triple Modular Redundancy</i>
AMFT	<i>Adaptive Middleware for Fault-Tolerance</i>
EPS	<i>Electrical Power System</i>
ADC	<i>Analog to Digital Converter</i>
PWM	<i>Pulse Width Modulation</i>
IO	<i>Input Output</i>
BIST	<i>Built-In Self Test</i>
SPI	<i>Serial Peripheral Interface</i>
RTC	<i>Real Time Clock</i>
LUT	<i>LookUp Table</i>
DFF	Flip Flop do tipo D

LISTA DE SÍMBOLOS

rad	Dose de radiação absorvida
t	Tempo
$R(t)$	Confiabilidade
$Q(t)$	Não confiabilidade
$z(t)$	Função de taxa de falha
λ	Taxa de falha
μ	Número esperado de reparos ao longo do tempo
$P_i(t)$	Probabilidade de o sistema estar no estado i no tempo t
M	Número de bits de dados
N	Número de bits totais
k	Número de bits de Hamming
ms	Tempo em milissegundos
mA	Corrente Elétrica em miliampéres
V	Tensão Elétrica

SUMÁRIO

1	INTRODUÇÃO	19
1.1	Objetivos	22
1.2	Foco da Pesquisa	23
1.3	Principais contribuições	24
1.4	Organização da tese	25
2	CENÁRIOS E SISTEMAS PARA COMPUTAÇÃO ESPACIAL	26
2.1	FPGAs do tipo COTS em ambientes espaciais	26
2.2	Confiabilidade de Sistemas de Computação	30
2.2.1	<i>Modelos de Confiabilidade</i>	32
2.2.2	<i>Cadeias de Markov</i>	34
2.2.3	<i>Failure Mode and Effect Analysis - FMEA</i>	36
2.3	Técnicas de Tolerância a falhas	37
2.3.1	<i>Circuito de detecção de SEL</i>	38
2.3.2	<i>Watchdog Timer</i>	39
2.3.3	<i>Redundância</i>	40
2.3.4	<i>Códigos Corretores de Erros</i>	41
2.4	Estratégias de Arquiteturas de OBC	44
2.4.1	<i>OBC 1</i>	47
2.4.2	<i>OBC 2</i>	49
2.4.3	<i>OBC 3</i>	50
2.4.4	<i>OBC 4</i>	53
2.4.5	<i>OBC 5</i>	55
2.4.6	<i>OBC 6</i>	56
2.4.7	<i>OBC 7</i>	58
2.4.8	<i>OBC 8</i>	58
2.5	Confiabilidade	59
2.5.1	<i>OBC 1</i>	61
2.5.2	<i>OBC 2</i>	62
2.5.3	<i>OBC 3</i>	63
2.5.4	<i>OBC 4</i>	65

2.5.5	<i>OBC 5</i>	67
2.5.6	<i>OBC 6</i>	67
2.5.7	<i>OBC 7</i>	68
2.5.8	<i>OBC 8</i>	68
2.6	Conclusão	69
3	ROBUST OBC: UM SISTEMA DE COMPUTAÇÃO DE BORDO TOLERANTE A FALHAS DE ALTA CONFIABILIDADE	71
3.1	Visão geral da arquitetura	71
3.1.1	<i>Sistema de Troca de Processador - PSS, Processor Switch System</i>	73
3.1.2	<i>Gerenciador de Falha - FM, Fault Manager</i>	77
3.1.3	<i>Sistema de Troca de Placa - BSS, Board Switch System</i>	79
3.1.4	<i>Sistema de Alimentação Inteligente - IPS, Intelligent Power System</i>	82
3.1.5	<i>Sistema de Comunicação do Processador e da Memória - PMCS, Processor and Memory Communication System</i>	84
3.2	Confiabilidade	86
3.3	Conclusão	88
4	RESULTADOS	90
4.1	Análise de falhas	90
4.2	Placa de Circuito Impresso do Robust OBC	94
4.3	Testes Funcionais do Robust OBC	95
4.3.1	<i>Teste do Sistema de Troca de Processador</i>	96
4.3.2	<i>Teste do Sistema de Troca de Placa</i>	102
4.3.3	<i>Teste do Sistema de Comunicação do Processador e da Memória</i>	107
4.3.4	<i>Resultado e Comparação de Confiabilidade</i>	109
4.4	Conclusão	112
5	CONCLUSÕES E PERSPECTIVAS	115
	REFERÊNCIAS	118
	APÊNDICES	123
	APÊNDICE A-ESQUEMAS ELÉTRICOS	123

1 INTRODUÇÃO

O uso de nano satélites pela indústria e academia têm crescido nos últimos anos devido ao baixo custo de projeto e lançamento, quando comparados com grandes satélites (THE CUBESAT STANDARD, 2021). Contudo, um dos grandes desafios em nano satélites é obter confiabilidade e alto desempenho no funcionamento destes sistemas, tendo em vista que estes, em sua maioria, são desenvolvidos utilizando componentes *Commercial Off-The-Shelf* (COTS) de baixo custo, especialmente para aplicações a partir de órbitas baixas. Estes componentes são sensivelmente susceptíveis aos efeitos da radiação ionizante presente no ambiente espacial, podendo, em alguns casos, causar mal funcionamento ou até mesmo paralisar todo o sistema do nano satélite quando estes são atingidos por partículas ionizantes. Neste sentido, se faz necessário desenvolver e aplicar novas técnicas de tolerância a falhas a nível de arquitetura de nano satélite para garantir confiabilidade mesmo na presença de falhas permanentes e transitórias nos componentes COTS.

O ambiente de radiação espacial possui partículas aprisionadas por magnetosferas planetárias em "cinturões", incluindo prótons, elétrons e íons mais pesados e partículas transitórias que incluem prótons e íons pesados de todos os elementos da tabela periódica. A radiação transitória é constituída de partículas de raios cósmicos galácticos, do inglês *Galactic Cosmic Radiation* (GCR) e partículas de eventos solares, tais como ejeções de massa coronal e erupções de chama. Estes dois tipos de erupções solares produzem prótons energéticos, partículas alfa, íons pesados e elétrons que estão em maior abundância do que os GCRs (BARTH *et al.*, 2003). Esses fenômenos são causadores de diversos tipos de falhas em componentes eletrônicos COTS os quais constituem a família de componentes utilizada em nano satélites baseados no padrão CubeSat. Este tipo de componente é bem mais susceptível à radiação do que os componentes que possuem proteção contra radiação. Os efeitos de radiação preponderantes nesses dispositivos mencionados são conhecidos como *Single Event Effects* (SEE) e *Total Ionizing Dose* (TID). Os efeitos SEE possuem classificações como *Single Event Upset* (SEU) e *Single Event Latchup* (SEL) que, assim como TID, são os que possuem maior incidência em componentes eletrônicos (CALDWELL, 1998).

Quando uma partícula altamente energizada deposita uma carga suficiente em regiões sensíveis do circuito integrado, ocorre uma alteração temporária do estado ou da operação do dispositivo como, por exemplo, inversão de bits em memórias. Este efeito é conhecido como SEU, enquanto no SEL, o mesmo fenômeno é capaz de causar um componente parasita dentro

do dispositivo através do qual uma corrente pode fluir danificando-o, caso exceda os seus limites máximo de tensão/corrente de operação (CALDWELL, 1998). Enquanto no TID, o componente eletrônico sofre degradação a longo prazo devido ao acúmulo de íons induzidos pela radiação. Quando isso ocorre, o componente poderá falhar completamente ou operar de forma indesejável (CALDWELL, 1998).

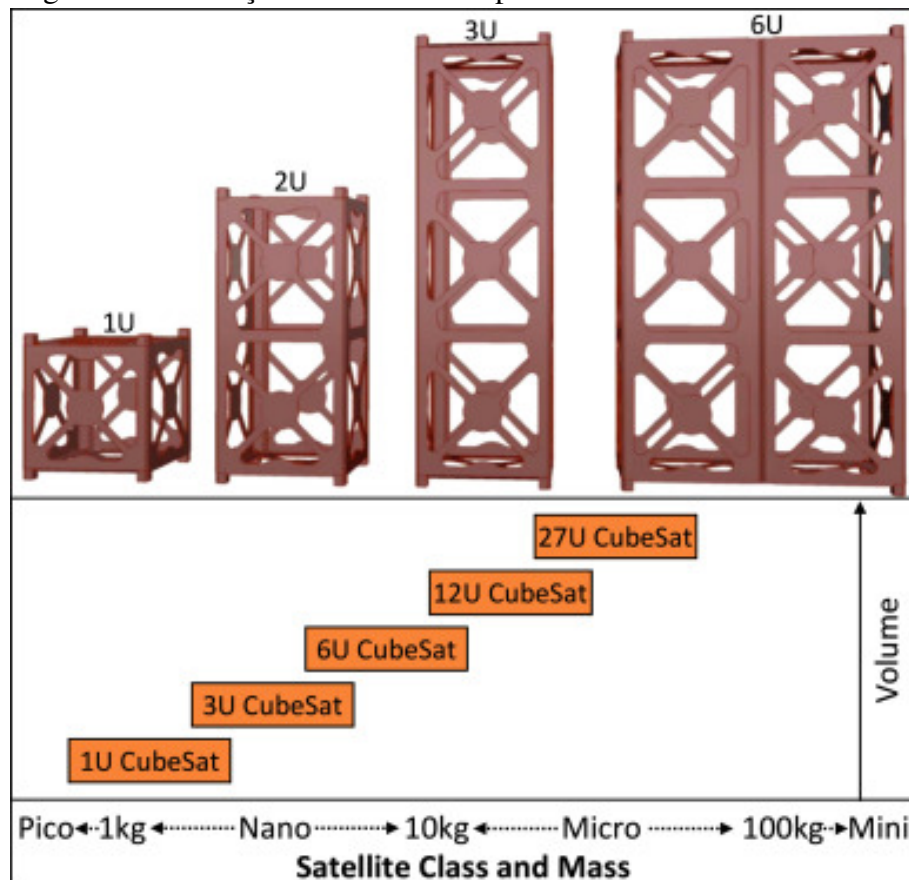
Por outro lado, grandes satélites não são tão susceptíveis a esses fenômenos, pois fazem uso de componentes com proteção a radiação, mais conhecidos como componentes *Radiation Hardening* (Rad-hard). Além disso, como são maiores e com um investimento mais alto, utilizam redundância física de circuitos integrados na maior parte de seus subsistemas eletrônicos (BOTMA *et al.*, 2013). Diferentemente, em nano satélites que fazem uso do padrão CubeSat, o tamanho, o peso e o custo são restritos, sendo necessário a utilização de técnicas de tolerância a falhas para mitigar eventuais problemas decorrentes dos fenômenos causados pela radiação (BOTMA *et al.*, 2013).

O padrão CubeSat tem sido muito utilizado em estudos e pesquisas em universidades, pois sua barreira de entrada para o espaço ficou mais acessível pelo fato dos custos serem mais baixos em comparação aos grandes satélites. Este padrão tem como vantagem a padronização de interfaces de comunicação entre os subsistemas do satélite e a estipulação de seu tamanho e peso (POGHOSYAN; GOLKAR, 2017). Existem vários tipos de missões em que um nano satélite, que faz uso do padrão CubeSat, é capaz de desempenhar, tais como coleta de dados ambientais, exploração do espaço profundo, Heliofísica: Clima Espacial, astrofísica, dentre outros (POGHOSYAN; GOLKAR, 2017). Além disso, o uso de um nano satélite para fins de pesquisa está sendo uma prática comum em diversos tipos de trabalhos acadêmicos como análises de sensores, qualidade de câmeras e os mais variados algoritmos como códigos corretores de erros.

Em 1999, nas universidades estaduais Politécnicas de Stanford e Califórnia, foi criado o padrão CubeSat que definiu um tamanho de 10x10x10cm e uma massa de 1,33Kg chamando esse cubo de 1U (POGHOSYAN; GOLKAR, 2017). O CubeSat 1U pode ser combinado com outros e assim gerar um satélite com maior tamanho como 3U, 6U, 12U, etc. A Figura 1 ilustra a combinação de CubeSats.

Além de massa e volume, o padrão CubeSat também definiu uma interface de comunicação entre os subsistemas existentes no satélite, a qual é encontrada facilmente em componentes eletrônicos COTS e em componentes Rad-Hard, como *Inter-Integrated Circuit*

Figura 1 – Formação de unidades do padrão CubeSat



Fonte: Poghosyan e Golkar (2017).

(I2C), *Controller Area Network* (CAN) e *Universal Asynchronous Receiver/Transmitter* (UART).

Componentes Rad-Hard são dispositivos resistentes à radiação e altas temperaturas (-55°C a 125°C). Eles desempenham o mesmo papel de componentes COTS, mas foram fabricados e testados para resistir a diferentes tipos de danos que acontecem no espaço ou em reatores nucleares. Como parte do processo de resistência à radiação, os componentes eletrônicos são blindados em uma camada de boro empobrecido e montados em substratos isolantes, ao invés de em pastilhas semicondutoras convencionais. Isto os torna capazes de suportar significativamente mais radiação do que os circuitos integrados COTS (MESSENGER, 1969).

Por causa do alto custo de engenharia e do processo de fabricação diferenciado, esse tipo de componente se torna caro e de difícil acesso, o que invalida, por parte das universidades em países emergentes como o Brasil, o desenvolvimento de algum subsistema de um nano satélite que faz uso do padrão CubeSat. Um subsistema um elemtno que é incorporado dentro do satélite para a realização de alguma tarefa específica. Exemplos de subsistemas: telemetria e telecomando, controle de atitude, fonte de alimentação, sensores, computador de bordo, etc (RAZZAGHI, 2012).

O computador de bordo, do inglês *On Board Computer* (OBC), é um sistema de computação que processa várias informações transmitidas ao satélite ou dos outros subsistemas embarcados. O principal objetivo do OBC é fornecer uma plataforma de *Command and Data Handling Subsystem* (CDHS) que faz interface com outros subsistemas de satélite e controla suas operações (RAZZAGHI, 2012). Em outras palavras, ele é o elemento central do satélite, podendo ser caracterizado como o cérebro do sistema.

Como o OBC gerencia todas as ações de um nano satélite, uma falha nele pode colapsar todo o sistema. Sua taxa de falhas pode ser relativamente alta e a maior porcentagem são em falhas aleatórias (49,2%) seguido por falhas de seu desenvolvimento (24,8%), falhas provenientes de ambiente desconhecido (21,4%) e falhas de operação (4,7%) (RAZZAGHI, 2012). Com isso, existe a necessidade de combater esses problemas através de um sistema de computação de bordo tolerante a falhas utilizando componentes COTS capaz de se recuperar, sem perder desempenho, desses defeitos descritos anteriormente.

1.1 Objetivos

Esta tese tem como objetivo principal a pesquisa e o desenvolvimento de um computador de bordo tolerante a falhas. Além disso, este projeto conta com os seguintes objetivos específicos:

- Utilização de elementos COTS em toda a arquitetura do computador de bordo, tendo em vista que componentes Rad-Hard são de difícil acesso principalmente por países em desenvolvimento;
- O interesse do INPE em construir/projetar um nano satélite nacional, mas que tenha um computador de bordo robusto;
- O desenvolvimento de uma arquitetura modular para recuperação de falhas de *hardware*;
- Testar o conceito de códigos corretores de erros aplicados em *Field-Programmable Gate Array* (FPGA) fazendo a interface entre um processador e uma memória *Static Random Access Memory* (SRAM);
- O desenvolvimento de um sistema baseado em lógica reprogramável que permita uma reconfiguração de todo o sistema em caso de falhas severas;
- Análise da confiabilidade do sistema de computação do OBC para falhas permanentes e transitórias.

1.2 Foco da Pesquisa

O foco dessa tese é na pesquisa, concepção, projeto, implementação e testes de uma arquitetura tolerante a falhas capaz de reduzir os erros pertinentes a computadores de bordo. Serão exploradas diversas técnicas de tolerância a falhas para combater TID e SEE em arquiteturas de OBCs. Para isso, um estudo de confiabilidade da arquitetura é feito, verificando pontos onde são necessárias redundâncias e onde haverá pontos únicos de falha sem que comprometa o desempenho deste computador de bordo.

Além disso, essa tese tem como foco a investigação de lacunas existentes nas arquiteturas de computadores de bordo existentes no estado da arte no sentido de propor soluções de tolerância a falhas nos principais elementos que compõem um OBC.

Um computador de bordo é composto por um elemento central, chamado de processador, memória de programa (Flash), memória de execução (SRAM), fonte de alimentação e interfaces de comunicação (RAZZAGHI, 2012). Assim, como o OBC é o elemento principal do satélite, o processador é o elemento principal do computador de bordo. Logo, há necessidade de aumentar a confiabilidade no sistema de processamento. Neste sentido, esse sistema deverá ser tolerante a falhas permanentes e transitórias dada sua importância no sistema.

Haja visto que a susceptibilidade à SEU em memórias SRAM é muito alta, será feito um estudo acerca dos principais códigos corretores de erros para combater possíveis inversões de bits que seriam causados pela radiação (RADAELLI *et al.*, 2005). Como este componente é imprescindível para um computador de bordo, será, também foco de estudo, a mitigação de problemas pertinentes a falhas permanentes.

Outro foco de pesquisa para enrobustecer a arquitetura do computador de bordo objeto dessa tese foi na solução de alimentação de energia. Os componentes eletrônicos necessitam de uma tensão de operação estável e confiável, para isso é proposto um sistema de alimentação que atenda a esses requisitos.

No próximo tópico serão levantados alguns questionamentos afim de propor contribuições científicas, no nível da fronteira do conhecimento, que possam aperfeiçoar as lacunas encontradas na literatura.

1.3 Principais contribuições

Após a realização de pesquisa nos principais veículos científicos, foram selecionados diversos computadores de bordo para nano satélites com as mais variadas arquiteturas tolerantes a falhas. Ao analisar, estudar e entender essas arquiteturas, algumas questões foram observadas com o intuito de contribuir para o preenchimento das lacunas encontradas. São elas:

1. Uma arquitetura tolerante a falhas utilizando componentes COTS é capaz de obter a mesma confiabilidade que uma outra utilizando componentes Rad-Hard?
2. Caso o processador sofra uma falha permanente, o que acontecerá com o OBC?
3. Não é possível prever qual o tipo de erro lógico irá acontecer na memória SRAM, logo, qual código corretor de erro mais adequado embarcar para mitigar essa falha? Somente um código é suficiente?
4. Qual a melhor maneira de aumentar a confiabilidade em um sistema de alimentação de energia?
5. É sabido que os computadores de bordo possuem componentes com maior importância que outros, além disso, como o OBC é de baixo custo, o que invalida o uso da técnica de redundância na maioria dos pontos únicos de falha, o que deverá ser feito caso o computador de bordo perca suas principais funções permanentemente?

No interesse de responder às citadas questões, foi verificado que alguns trabalhos tem proposto aumentar a tolerância a falhas em sistemas computacionais para aplicações espaciais, conforme detalhado no capítulo 2 (inclusive com uma comparação profunda das arquiteturas no estado da arte com a proposta nessa tese). Contudo, muitos desses trabalhos ou apenas focam na proteção da memória SRAM, utilizando códigos corretores de erro, ou utilizam redundância ativa dos processadores principais. Além, claro, das arquiteturas que utilizam componentes eletrônicos Rad-hard, os quais são de alto custo de implementação, e de difícil, ou quase impossível, acesso pelos países em desenvolvimento industrial, como é o caso do Brasil. Então, uma solução completa que tenha em sua arquitetura o uso de componentes COTS, técnicas de tolerância a falhas no sistema de processamento, proteção física e lógica da memória, sistema de alimentação robusto e inteligente, além de redundância de OBC é de fundamental importância no contexto mundial. Neste sentido, é proposto nesta tese o Robust OBC, uma arquitetura robusta e de alta confiabilidade para computadores de bordo de um nano satélite que faça uso do padrão CubeSat.

Em resumo, as principais contribuições desta tese são:

1. Arquitetura baseada em redundância de OBCs;
2. Redundância de processadores utilizando um sistema de troca inteligente;
3. Sistema dinamicamente reconfigurável para proteção de memória SRAM;
4. Implementação de um gerenciador de falha para falhas permanentes e transitórias;
5. Arquitetura projetada para realização de chaveamento de contexto na comutação dos processadores e na comutação dos computadores de bordo.

1.4 Organização da tese

Esta tese está organizada da seguinte forma: no capítulo 2 será feita a pesquisa e o estudo no estado da arte dos diversos cenários e sistemas para computação em ambientes espaciais analisando o uso de FPGAs nesse tipo de ambiente, a confiabilidade desse modelo de sistema, as técnicas de tolerância a falhas pertinentes e as estratégias utilizadas nos OBCs. O capítulo 3 mostrará todo o desenvolvimento do Robust OBC focando na visão geral do sistema e baixando o nível adentrando nos detalhes das técnicas utilizadas para preencher as lacunas percebidas na pesquisa científica, além disso será visto o modelo de confiabilidade do Robust OBC juntamente com sua expressão matemática. A análise de falhas, o *setup* de teste, os resultados obtidos e a comparação das confiabilidades entre os OBCs visto no capítulo 2 e o Robust OBC serão mostrados no capítulo 4. Por fim, no quinto capítulo, será feita a conclusão e o levantamento de trabalhos futuros.

2 CENÁRIOS E SISTEMAS PARA COMPUTAÇÃO ESPACIAL

Diversos são os desafios quando se trata de computação embarcada para exploração espacial, o principal deles está relacionado aos efeitos da radiação ionizante, presente em maiores quantidades nas orbitas espaciais, nos componentes de eletrônica digital embarcados. Neste capítulo é feita uma breve descrição desses efeitos da radiação nos componentes eletrônicos digitais do tipo COTS, principalmente naqueles baseados em FPGAs. Ainda nesse capítulo, é abordado o conceito de confiabilidade de sistemas digitais, com olhos para técnicas de tolerância a falhas que possam prover maior robustez contra os efeitos da radiação em sistemas computacionais modernos. Além disso, é realizada uma metodologia para a seleção dos OBCs que irão ser estudados e analisados as estratégias utilizadas em suas concepções de arquitetura com objetivo de encontrar lacunas e melhorias para o desenvolvimento do Robust OBC, objeto dessa tese. Ainda na análise desses OBCs, serão implementados os modelos de suas arquiteturas para a obtenção da expressão matemática da confiabilidade no intuito de comparar com o Robust OBC. Por fim, a conclusão que irá encerrar o capítulo.

2.1 FPGAs do tipo COTS em ambientes espaciais

Como comentado no capítulo 1, os componentes eletrônicos COTS são susceptíveis à radiação espacial. Os efeitos causados por esses fenômenos podem ser caracterizados como falhas transitórias ou permanentes. Falhas permanentes geralmente são causadas por defeitos físicos no *hardware*, como curtos-circuito, conexões danificadas ou células danificadas em uma memória. Uma falha transitória acontece por um curto período de tempo. Devido à sua curta duração, as falhas transitórias são freqüentemente detectadas por meio dos erros que resultam de sua propagação. Caso esse tipo de falha se torne ativa periodicamente, logo, ela é denominada de falha intermitente. Uma falha física pode se manifestar como um erro, como em uma inversão de bit, ou o efeito da falha pode ser mascarado e não se manifestar como qualquer erro. Da mesma forma, um erro pode ser mascarado e pode resultar em um comportamento incorreto visível ao usuário chamado de falha. As falhas incluem cálculos incorretos e travamento do sistema. (DUBROVA, 2013).

O TID é um causador de falhas permanente oriundo de um depósito contínuo de radiação ionizante no dispositivo. Esse depósito é inevitável e faz com que o componente se degrade lentamente ao longo do tempo. Uma das características dos elementos Rad-Hard é ser

classificado pela quantidade de depósitos que ele pode suportar, logo, as formas de solucionar esse problema é utilizando um componente com essa característica ou trocando o dispositivo afetado (HANE, 2012).

O SEL é um outro evento que, caso não identificado e nem realizado nenhuma intervenção, poderá se tornar uma falha permanente. Isto acontece quando a radiação ionizante ativa transistores parasitas no substrato de silício, isso faz com que uma corrente flua danificando o componente. Neste caso, para evitar danos permanentes no componente se faz necessário a remoção da alimentação do mesmo enquanto persistir o curto-circuito induzido pelo SEL, pois essas correntes podem gerar superaquecimento e algumas partes do componente poderão queimar (HANE, 2012).

As falhas transitórias provenientes de *Single Event Transient* (SET), acontecem quando uma partícula de alta energia se choca com o dispositivo provocando a ionização de alguns dos átomos de silício. Neste caso, se houver carga suficiente concentrada na área para reverter o estado de um sinal lógico digital no sistema, o efeito é chamado de SET. Um SET provavelmente resultará em uma falha breve antes que o excesso de carga se dissipe, a menos que o novo estado do sinal seja capturado e retido por um dispositivo de armazenamento (flip-flops). Um SET que é capturado desta forma é identificado como um SEU e pode ter um efeito bastante indesejável, pois pode alterar o valor originalmente contido nos flip-flops. Contudo, um SEU pode ser corrigido com uma reinicialização do sistema que restaura todos os flip-flops para um estado conhecido. Por outro lado, os SEUs que não podem ser reparados pela reinicialização são chamados de *Single Event Functional Interrupt* (SEFI) que afetam sinais de controle e máquinas de estado, colocando o dispositivo em um estado indefinido (HANE, 2012).

Um dos componentes eletrônicos com maior susceptibilidade a SEU é a memória SRAM (MAQBOOL, 2006). Isso faz com que se tenha uma atenção especial à proteção desse dispositivo dada sua importância para o funcionamento do computador de bordo. Duas técnicas podem ser utilizadas para mitigar esses possíveis eventos que causam mau funcionamento das memórias SRAMs. São elas: *Error Detecting Codes* (EDC) e *Error Detection And Correction* (EDAC). O sistema que faz uso de um EDC gera, transmite e armazena palavras codificadas, ou seja, os dados originais mais os bits do código de detecção de erro (MAQBOOL, 2006). Códigos de paridade e códigos de redundância cíclica são exemplos de EDCs (DUBROVA, 2013). Os EDACs são usados para proteger os dados contra erros que podem ocorrer em células de armazenamento ou canais de transmissão. Quando a unidade de processamento solicita dados da

memória, todos os bits são lidos pelo EDAC, que realiza a detecção e correção de erro, e entrega os bits corrigidos para a unidade de processamento. Todavia, o EDAC não escreve novamente os dados corrigidos de volta na memória. Então, para manter a memória sem acúmulo de erros, a mesma deverá ser periodicamente lida e escrita. Este procedimento é chamado de *washing* ou *scrubbing* (MAQBOOL, 2006). Os códigos EDACs mais conhecidos são: Hamming, Hamming Estendido e Reed-Solomon (DUBROVA, 2013).

Há duas maneiras de se implementar esses códigos corretores de erro, via *software* e via *hardware*. A implementação por *software* faz com que o computador de bordo utilize a maior parte de seu processamento com a execução desse programa em detrimento às outras tarefas que ele deveria estar executando. Já a implementação em *hardware*, por ser embarcada, em sua maioria, em dispositivos lógicos programáveis e poder ser um circuito dedicado para a execução de códigos corretores de erros, é bem mais rápida no desempenho dessa tarefa que a implementação via *software*. Uma desvantagem é que isso pode consumir mais espaço na Placa de Circuito Impresso (PCI) além do aumento do consumo de potência elétrica pela adição deste componente eletrônico no projeto (DUBROVA, 2013).

O dispositivo lógico programável que mais vem sendo utilizado em sistemas de computação tolerante a falhas é a FPGA (KUWAHARA, 2010). A importância do uso de uma FPGA nesse trabalho veio da necessidade de proteção da memória SRAM contra SEUs, abrindo espaço, também, para a utilização em diversos outros circuitos como forma de implementar técnicas de tolerância a falhas para dar mais robustez na arquitetura proposta nesta tese. Porém, a FPGA COTS, do ponto de vista de confiabilidade, torna-se um ponto único de falhas *Single Point Failure* (SPF).

Diferentemente dos trabalhos (MERL; GRAHAM, 2016), (GEORGE; WILSON, 2018), (ZIN TECHNOLOGIES, 2021) e (NOELDEKE *et al.*, 2021), nos quais as FPGAs utilizadas são do tipo Rad-Hard, no Robust OBC é utilizada uma FPGA COTS do fabricante Microsemi que possui algumas características de tolerância a falhas em sua construção. Ela é baseada em Flash e foi desenvolvida com tecnologia de 65nm.

Existem, principalmente, três tipos diferentes de FPGAs: SRAM, Flash e FPGAs Antifuse. A primeira FPGA foi baseada na tecnologia SRAM do fabricante Xilinx. As FPGAs SRAM são voláteis e o código embarcado deve ser armazenado em memória externa não volátil fazendo com que o mesmo seja carregado na inicialização do sistema (KUWAHARA, 2010).

Já a FPGA Antifuse é do tipo *One-Time Programmable* (OTP), e uma vez progra-

mada, não é caracterizada como não-volátil. O tamanho da área das FPGAs Antifuse é bem inferior em comparação com as FPGAs SRAM (KUWAHARA, 2010).

Por último, existem as FPGAs Flash que são reconfiguráveis e não voláteis. Porém, devido à sua complexidade de fabricação, é difícil construí-las de forma barata e com uma alta integração. Com isso, seu mercado não é tão grande, pois não possuem uma quantidade de lógica elevada e não executavam operação de alta velocidade (KUWAHARA, 2010).

As FPGAs Antifuse são tolerantes a TID e SEE, porém esses fenômenos são causadores de problemas nos demais tipos de FPGA. Os efeitos TID em FPGAs SRAM e Flash são caso de investigação de pesquisadores a bastante tempo, tendo como objeto de estudo a família Virtex do fabricante Xilinx para SRAM e ProASIC do fabricante Microsemi para a Flash, no qual foi verificado que o componente suporta 300 krad no caso da SRAM e até 20 Krad para o caso da Flash. Para missões em órbita terrestre baixa, do inglês *Low Earth Orbit* (LEO), esse valor de 300 Krad é suficiente para caracterizar a FPGA como imune, porém pode-se afirmar que as FPGAs Flash são vulneráveis a esse fenômeno (KUWAHARA, 2010).

Para os efeitos SEE, as FPGAs SRAM são vulneráveis enquanto as FPGAs Flash são imunes para a maioria dos tipos desse efeito (KUWAHARA, 2010).

Com base na análise da pesquisa realizada acerca dos tipos de FPGAs e de sua tolerância aos fenômenos existentes na radiação, assim como na pesquisa em diversos fabricantes de FPGA para selecionar o dispositivo que atendesse aos requisitos do OBC objeto dessa tese, foi escolhida a FPGA Flash M2S010-VFG400I da família SmartFusion2.

Diversos trabalhos avaliaram e testaram esta FPGA da família SmartFusion2 para os mais variados efeitos que simulam os fenômenos de radiação. Em (DSILVA *et al.*, 2015), a SmartFusion2 foi testada em ambientes com íons pesados e radiação de nêutrons para verificação de tolerância a falhas para SEE. Os testes mostraram que essa família funciona de maneira satisfatória nesse tipo de ambiente, tendo somente que mitigar os problemas pertinentes aos dados em flip-flops e nos blocos de SRAM através de códigos corretores de erros. Enquanto em (REZZAK *et al.*, 2014), a SmartFusion2 foi testada para verificação de tolerância a TID. Essa verificação foi apresentada utilizando três parâmetros: a degradação no atraso de propagação versus a dose total aplicada, a corrente de alimentação no *core* da SmartFusion2 e a degradação V_t dos comutadores da FPGA Flash. Com a injeção de raios gama de 10 Krad até 100 Krad, os principais sintomas desses testes foram no atraso da velocidade de funcionamento e no aumento de consumo na corrente do *core* da FPGA. O funcionamento ocorre normalmente, porém com os

problemas acima citados.

Dessa forma, a opção pela FPGA Flash SmartFusion2 foi definida tendo em vista o fato dela ser imune a SEU e a SEL e possuir resistência aos efeitos de TID. Contudo, como essa tese se propõem a prover um OBC robusto para ser utilizado em aplicações espaciais, somente essas características de construção da FPGA não são suficientes para garantir o bom funcionamento do computador de bordo e conseqüentemente do nano satélite. Logo, para mitigar as lacunas vigentes na seleção da FPGA Flash COTS, uma arquitetura robusta e de alta confiabilidade foi projetada e será discutida no capítulo 3.

No item a seguir, é discutido a confiabilidade em sistemas de computação para dar respaldo e consistência aos argumentos utilizados para o desenvolvimento da arquitetura proposta nesta tese.

2.2 Confiabilidade de Sistemas de Computação

Quando se trata de sistemas de computação tolerantes a falhas, um dos principais parâmetros a ser considerado é a Confiabilidade. Isto é importante, pois é através da análise da confiabilidade que podemos verificar qual deve ser o melhor caminho a ser seguido na implementação da arquitetura de um sistema computacional. Alta confiabilidade é uma métrica importante quando estamos falando de um sistema cujo as falhas possam causar efeitos catastróficos como é o caso de falhas aviônicas (SORIN, 2009). Isto é, a confiabilidade é necessária quando um sistema opera sem interrupções ou quando o sistema é remoto e impossível de se ter acesso para realizar manutenções como, for exemplo, um satélite (DUBROVA, 2013). A confiabilidade de um sistema no momento t é a probabilidade de que o sistema esteja operando corretamente do tempo zero até o tempo t , dado que o sistema estivesse funcional no início do tempo, ou seja, a confiabilidade é uma função do tempo (DUBROVA, 2013).

Em resumo, a confiabilidade indica a probabilidade do correto funcionamento de um determinado sistema por um período de tempo, já a não confiabilidade indica a probabilidade de o sistema falhar em um intervalo de tempo (DUBROVA, 2013). A confiabilidade e a não confiabilidade são definidos como $R(t)$ e $Q(t)$ respectivamente e são relacionados como (DUBROVA, 2013)

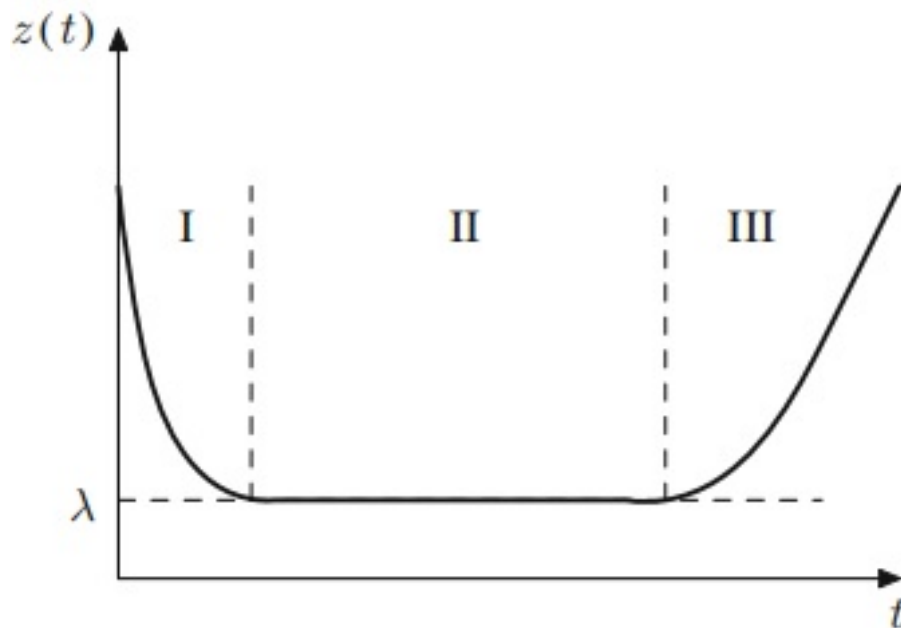
$$Q(t) = 1 - R(t). \quad (2.1)$$

As falhas possuem uma tendência de aumentar sua probabilidade de ocorrência ao

longo do tempo de funcionamento do sistema. Isso faz com que a taxa de confiabilidade diminua nesse mesmo período de tempo.

A taxa de falhas é definida como o número esperado de falhas por unidade de tempo. A Figura 2 demonstra um gráfico do padrão da taxa de falhas em projetos de *hardware*. A curva ilustrada nesse gráfico é feita durante o período de vida útil de um sistema e possui três fases: (I) mortalidade infantil, (II) vida útil, e (III) desgaste. No primeiro momento, a taxa de falhas diminui acentuadamente, dado as falhas freqüentes de componentes defeituosos em que esses defeitos de fabricação foram desconsiderados durante os testes em seu tempo de construção. Depois de um certo período, a taxa de falhas tende a se estabilizar e permanece constante. Finalmente, com o desgaste dos componentes eletrônicos ou mecânicos, a taxa de falhas aumenta (DUBROVA, 2013).

Figura 2 – Curva de taxa de falhas em projetos de *hardware* (*Bath Tube*).



Fonte: Dubrova (2013).

Enquanto o sistema está na fase de vida útil, a função de taxa de falha $z(t)$ é caracterizada como uma constante λ . Logo, a confiabilidade do sistema diminui exponencialmente com o tempo e é expressa conforme a equação (DUBROVA, 2013)

$$R(t) = e^{-\lambda t}. \quad (2.2)$$

Como a confiabilidade é a probabilidade de o sistema estar em operação normal e contínua em um determinado período de tempo, o tempo médio até o sistema falhar, do inglês

Mean Time To Failure (MTTF), e o tempo médio entre as falhas que ocorreram, do inglês *Mean Time Between Failure* (MTBF), estão relacionados entre si (YANG *et al.*, 2017).

A relação entre MTTF e MTBF é dada através do tempo médio de reparo, do inglês *Mean Time to Repair* (MTTR). O MTTR é representado em termos da taxa de reparo μ , que é o número esperado de reparos ao longo do tempo, então (DUBROVA, 2013)

$$MTTR = \frac{1}{\mu}. \quad (2.3)$$

MTTF é o tempo empregado até o momento que ocorre a falha do sistema e possui uma expressão matemática que modela o tempo de operação do sistema irreparável antes da falha, com isso (YANG *et al.*, 2017),

$$MTTF = \int_0^{\infty} R(t)dt = \int_0^{\infty} e^{-\lambda t} dt = \frac{1}{\lambda}. \quad (2.4)$$

Logo, o tempo médio entre as falhas é (YANG *et al.*, 2017):

$$MTBF = MTTF + MTTR. \quad (2.5)$$

2.2.1 Modelos de Confiabilidade

Para se obter um sistema de computação tolerante a falhas, é necessário estudar, analisar e comparar a confiabilidade do sistema com objetivo de aumentar a sua robustez. Para isso, existem modelos de interconexões de confiabilidade que auxiliam no desenvolvimento de uma arquitetura mais adequada para o produto final desejado (YANG *et al.*, 2017). São eles:

- Modelo Serial
- Modelo Paralelo
- Modelo de Múltipla Redundância

No modelo em Série, todos os componentes devem funcionar para que o sistema completo funcione. A confiabilidade desse modelo tende a diminuir com o aumento de componente em série no sistema (YANG *et al.*, 2017). A Figura 3 ilustra o modelo de um sistema em série.

Figura 3 – Modelo de um sistema em Série.



Fonte: O autor.

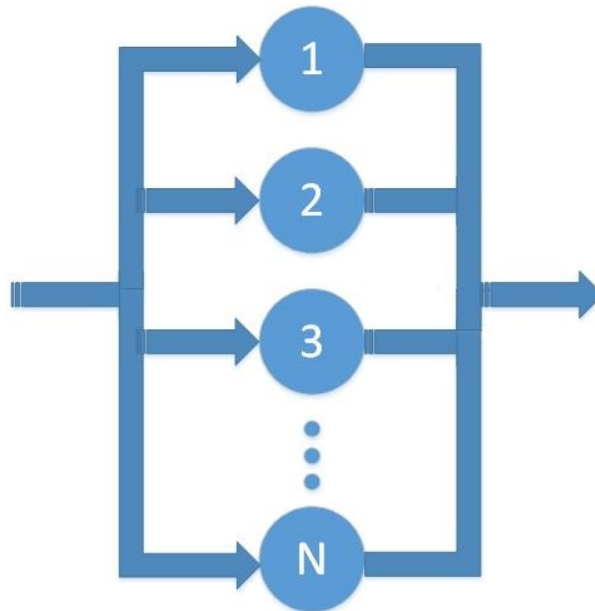
A equação matemática que expressa este modelo é:

$$R(t) = \prod_{i=1}^n R_i(t). \quad (2.6)$$

$R(t)$ é a confiabilidade do sistema, $R_i(t)$ é a confiabilidade de cada componente e n é o total de componentes do sistema em série (YANG *et al.*, 2017).

Uma forma de aumentar a confiabilidade está no uso de um modelo em paralelo que, diferentemente do modelo em série, a mesma aumenta conforme a inclusão de novos componentes e caso haja alguma falha, não afetará os componentes sobressalentes. Com isso, o sistema completo só falhará caso todos os componentes falhem (YANG *et al.*, 2017). A Figura 4 ilustra o modelo de um sistema em paralelo.

Figura 4 – Modelo de um sistema em Paralelo.



Fonte: O autor.

A expressão matemática do modelo em paralelo é:

$$R(t) = 1 - \prod_{i=1}^n (1 - R_i(t)). \quad (2.7)$$

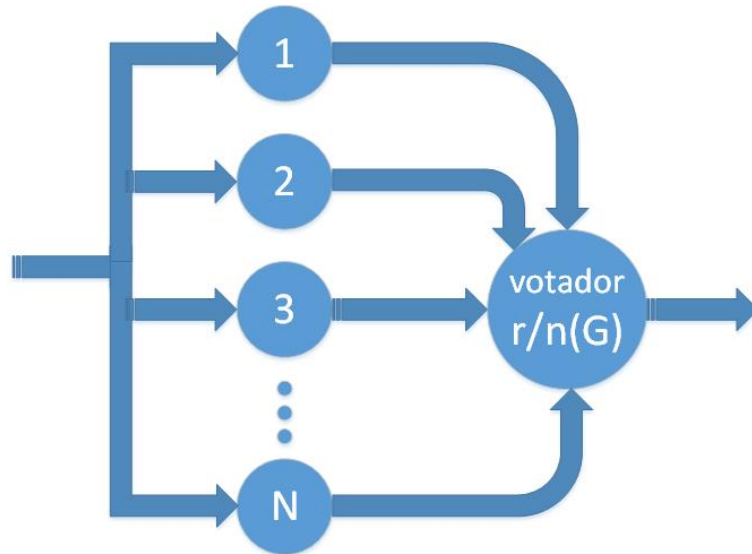
As variáveis são as mesmas descritas no modelo em série (YANG *et al.*, 2017).

Já o modelo de Múltipla Redundância funciona como um modelo de cópia de segurança. Este modelo é constituído por n componentes e um votador, e sua confiabilidade é definida como $r/n(G)$ conforme a Figura 5.

O modelo de Múltipla Redundância possui a expressão matemática conforme:

$$R(t) = R_m \sum_{i=1}^n C_n^i R(t)^i (1 - R(t))^{(n-i)}. \quad (2.8)$$

Figura 5 – Modelo de Múltipla Redundância.



Fonte: O autor.

$R(t)$ é a confiabilidade do sistema, $R(t)^i$ é a confiabilidade de cada um dos componentes do sistema (idênticos para cada componente) e R_m é a confiabilidade do votador (YANG *et al.*, 2017).

No próximo item, será descrito o processo de Cadeia de Markov. Esse processo é importante para a análise de confiabilidade em sistemas de computação tolerante a falhas em que o estado atual do sistema não tenha dependência do estado anterior.

2.2.2 Cadeias de Markov

Os sistemas de computação tolerantes a falhas para aplicações críticas possuem diversos elementos que dificultam sua análise no que diz respeito a confiabilidade. Os projetos desenvolvidos para obter uma alta confiabilidade geralmente possuem grandes níveis de redundância, reconfiguração dinâmica do sistema e técnicas complexas de recuperação de falhas e erros (DUGAN *et al.*, 1993).

Existem várias técnicas para a análise de confiabilidade em sistemas de computação tolerantes a falhas para aplicações críticas. Além de prever a confiabilidade do sistema para um tempo específico de uma determinada missão, essas estratégias de análises de falhas facilitam na seleção de várias técnicas de tolerância a falhas ou podem ser usadas para comparar arquiteturas alternativas para um sistema ainda em fase de desenvolvimento (DUGAN *et al.*, 1993).

Uma estratégia de análise de falhas em sistemas de computação tolerante a falhas para aplicações críticas é a Cadeia de Markov. Ela apresenta uma modelagem alternativa que é

flexível o suficiente para modelar qualquer sistema dinâmico. Esses comportamentos dinâmicos incluem recuperação transitória, erros intermitentes e dependência de sequência (DUGAN *et al.*, 1993).

Cadeia de Markov é um processo estocástico que possui a propriedade markoviana. Um processo estocástico é dito markoviano se os estados anteriores do processo são irrelevantes para a predição dos próximos estados, porém o estado atual deve ser conhecido. Cadeias de Markov contínuas no tempo são a classe mais importante de processos de Markov para avaliação de confiabilidade (DUBROVA, 2013).

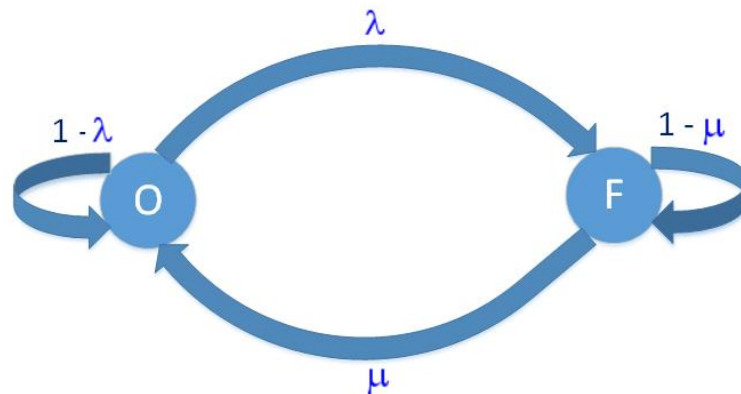
O objetivo da modelagem da confiabilidade ao longo do tempo pelo processo de Markov é calcular a probabilidade de o sistema estar no estado i no tempo t ($P_i(t)$). Uma vez que esse estado é conhecido, a confiabilidade do sistema pode ser calculada como a soma das probabilidades de todos os estados funcionais. Dado que o estado 1 seja o estado em que todos os componentes estão operacionais e assumindo que em $t = 0$ o sistema está no estado 1, logo $P_1(0) = 1$. Como o sistema só pode estar em apenas um estado, $P_i(0) = 0$, então,

$$\sum_{i \in OUF} P_i(t). \quad (2.9)$$

Isto é o somatório de todos os estados operacionais (O) e estados com falha (F) possíveis (DUBROVA, 2013).

A Cadeia de Markov é representada pelo diagrama de transição de estados que tem como função analisar os dois estados de um componente que podem ser operação ou falha. A probabilidade de mudar do estado O para F é P_{1-2} ou λ e a probabilidade de mudar do estado F para O é P_{2-1} ou μ . A Figura 6 ilustra uma cadeia de Markov.

Figura 6 – Diagrama de transição de estados da Cadeia de Markov.



Fonte: O autor.

O diagrama de transição de estados pode ser representado por uma Matriz de Transição Estocástica (\hat{P}). As probabilidades de transição de estados são armazenados nas células da matriz, sendo que essas probabilidades podem ou não variar com o tempo. Se $\hat{P}(t) = \hat{P}$, a matriz é considerada homogênea, ou seja, é uma matriz estacionária e suas probabilidades de transição são independentes do tempo (BACELO, 2016), representada na equação 2.10:

$$P = \begin{pmatrix} P_{1,1} & P_{1,2} & \cdots & P_{1,n} \\ P_{2,1} & P_{2,2} & \cdots & P_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ P_{m,1} & P_{m,2} & \cdots & P_{m,n} \end{pmatrix}, \quad (2.10)$$

em que P_{ij} representa a probabilidade de transição do estado i para o estado j .

A evolução de uma cadeia de Markov ao longo do tempo é chamada de diagrama de Árvore ou Árvore de Falhas. Árvores de falha são frequentemente usadas para análise de confiabilidade de sistemas críticos. Os modelos de árvore de falhas são bem disseminados e seus métodos bastante utilizados, porém a análise exata de árvores de falhas com muitos eventos básicos pode ser bem complexo, tanto em termos de desenvolvimento do modelo quanto na resolução do modelo uma vez desenvolvido (DUGAN *et al.*, 1993).

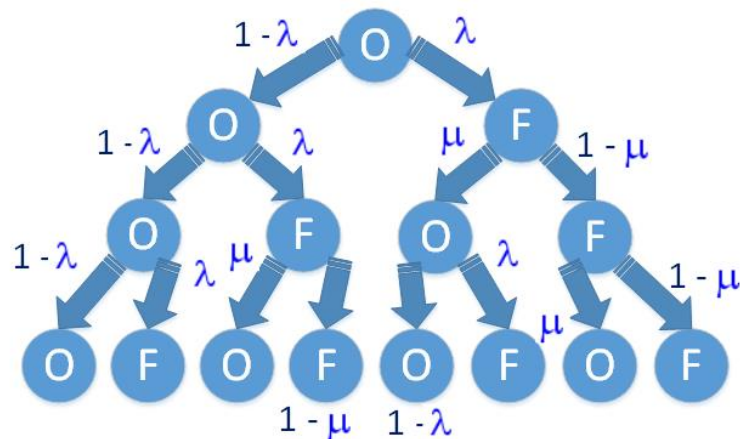
As árvores de falha descrevem possíveis sequências de eventos que podem levar a uma falha do sistema. A dependência operacional entre os componentes de um sistema é representada usando portas lógicas booleanas (por exemplo, AND, OR). Os componentes são representados como nós de entrada na árvore, fornecendo entradas para as portas (DUGAN *et al.*, 1993). A Figura 7 ilustra o diagrama de árvore referente ao diagrama de transição de estado da Figura 6.

Um exemplo de um diagrama de árvore é um método chamado de análise para modos e efeitos de falhas, do inglês *Failure Mode and Effect Analysis* (FMEA).

2.2.3 *Failure Mode and Effect Analysis - FMEA*

FMEA é uma técnica utilizada para prover a confiabilidade de um produto ou sistema durante a fase de projeto. A técnica consiste basicamente em relacionar um grupo de atividades com objetivo de detectar possíveis falhas e avaliar os efeitos das mesmas para o projeto. A partir dessas possíveis falhas, são identificadas algumas ações a serem tomadas para mitigar ou reduzir a probabilidade de que as mesmas ocorrem. As ações também devem aumentar a probabilidade

Figura 7 – Exemplo de um Diagrama de Árvore.



Fonte: O autor.

de detecção dessas falhas, para que os produtos não saiam da fábrica para o consumidor com possíveis não conformidades (BACELO, 2016).

O FMEA de *hardware* pode identificar todos os modos de falha possíveis. A partir da análise dos efeitos da falha, a fragilidade do sistema pode ser encontrada e medidas devem ser tomadas e implementadas para aumentar a confiabilidade. Todos os modos de falha de cada componente devem ser analisados, para garantir que não exista nenhuma falha inaceitável no sistema (YANG *et al.*, 2017).

Nessa tese, o conceito de FMEA será aplicado na realização de um diagrama de falhas no qual serão descritos as principais falhas que possam ocorrer nos componentes críticos do sistema de computação e os caminhos e tomadas de decisão que serão percorridos para que essas falhas não causem falhas catastróficas e inviabilizem a missão ao qual o nano satélite está submetido.

2.3 Técnicas de Tolerância a falhas

Quando se usa componentes COTS em aplicações críticas, como por exemplo em aplicação aeroespacial, é necessário utilizar algumas técnicas que mitiguem eventuais falhas pertinentes ao ambiente hostil onde esses componentes estão submetidos. Essas técnicas são conhecidas como técnicas de tolerância a falhas, podendo ser empregadas tanto em *hardware* como em *software* e variam de acordo com as necessidades desejadas (LIM, 2009). No caso de um OBC para nano satélites que faz uso do padrão CubeSat, as técnicas são variadas e dependem da missão que ele irá desempenhar.

O OBC pode ser projetado ou para ter uma alta confiabilidade, não permitindo falhas,

ou para alto desempenho com um tempo mínimo de inatividade. No primeiro caso, a tolerância a falhas é importante para mascarar as falhas encontradas e garantir que não haja interrupção nas operações. Já no segundo caso, as falhas são tratadas e então o sistema é reconfigurado para manter o nível de desempenho necessário. Neste processo de reconfiguração, os serviços podem ser interrompidos momentaneamente trazendo prejuízos para as operações do sistema (LIM, 2009).

Quanto maior o uso das técnicas de tolerância a falhas, maior será o custo associado ao produto final, tais como área, potência, desempenho e o preço associado a implementação física dos circuitos eletrônicos. Como as aplicações de CubeSat são, geralmente, de baixo custo, o ideal é que haja sempre um compromisso na relação custo/benefício no que diz respeito a escolha da técnica de tolerância a falhas a ser utilizada nesse contexto.

Diversas são as técnicas de tolerâncias a falhas utilizadas para as mais variadas aplicações. Neste trabalho serão discutidas as seguintes quatro técnicas: Circuito de detecção SEL; *Watchdog Timer*; Redundância; EDAC.

2.3.1 Circuito de detecção de SEL

A técnica de tolerância a falhas contra SEL foi bastante difundida na literatura ao longo dos anos (SHOGA; BINDER, 1986), e ainda é muito utilizada nos dias de hoje. Diversos trabalhos, como por exemplo em (NICOLAIDIS, 2006), inovaram no uso dessa técnica oferecendo uma maior confiabilidade através do uso combinado de EDAC com circuito de detecção de *latchup*, que pode variar de acordo com a aplicação que está sendo submetido.

A necessidade da concepção dessa técnica se deu pelo fato do avanço tecnológico em componentes *Complementary Metal-Oxide-Semiconductor* (CMOS) e, conseqüentemente, sua susceptibilidade a efeitos causados por ions pesados ou prótons provenientes de GCR e partículas de eventos solares (LAYTON *et al.*, 1997).

Essa técnica consiste em limitar o consumo de corrente do componente eletrônico que se deseja proteção. Após isso, é feita uma detecção no aumento de corrente durante o evento de SEL, verificando se a corrente está acima do limite predefinido. Essa definição do limite da corrente varia de acordo com o componente em questão e o valor é encontrado na folha de dados do fabricante do componente. Em alguns casos, esse valor também é encontrado através de experimentos em laboratório. Quando esse limite é atingido, é necessário retirar a tensão de alimentação do dispositivo e manter ele desligado por um tempo predefinido até que todo o

efeito tenha sido removido para, então, inserir a tensão de alimentação novamente (LAYTON *et al.*, 1997).

Em aplicações de CubeSat, a utilização dessa técnica é imprescindível dado que ela irá realizar proteção nos mais variados componentes eletrônicos críticos, que, caso falhem permanentemente devido ao aumento de sua corrente, podem inviabilizar a missão. Um exemplo disso é a memória SRAM, que além de ter um papel importante dentro do funcionamento de um OBC, ela também apresenta alta susceptibilidade a este tipo de efeito e por isso, necessita ter proteção contra SEL (SECONDO *et al.*, 2016).

2.3.2 *Watchdog Timer*

Um *Watchdog timer* é um dispositivo que monitora outro dispositivo em busca de seus sinais de funcionamento. Uma das mais variadas aplicações de um *Watchdog timer* é na vigilância de um processador. Um processador sempre solicita dados de uma memória através de um barramento, e se nenhuma solicitação tiver sido observada por um determinado tempo que exceda um limite de tempo predefinido, o *Watchdog timer* irá informar que ocorreu um erro. A verificação do bom funcionamento de um processador, através de um *Watchdog timer*, é mais eficaz e simples do que realizar todas as operações do processador para constatar que o mesmo encontra-se em perfeita condições. Além disso, o circuito eletrônico utilizado para contruir um *Watchdog timer* é simples e de baixo custo quando comparados com outros elementos de um sistema computacional, como por exemplo o processador. Com isso, um *Watchdog timer* pode, portanto, ser muito mais barato do que um processador redundante (SORIN, 2009).

Assim como os circuitos contra SEL, a técnica de *Watchdog timer* é antiga (MAHMOOD; MCCLUSKEY, 1988), porém ainda é bastante utilizada dada a sua eficiência comprovada em diversos trabalhos, tais como (OHLSSON *et al.*, 1992), (NAMJOO; MCCLUSKEY, 1995) e (BENSO *et al.*, 2003).

O uso da técnica de *Watchdog timers* em OBCs tem se mostrado muito eficiente conforme descrito em (BUSCH; SCHILLING, 2013). Essa técnica tem se popularizado neste tipo de aplicação pois ao mesmo tempo em que aumenta a tolerância a falhas dos OBCs, ela oferece um baixo custo de implementação.

2.3.3 Redundância

Redundância é o acréscimo de informações, recursos ou tempo além do necessário para que um sistema opere em condições normais (PRADHAN, 1996). A redundância é importante em situações que um determinado equipamento não pode ser reparado como, por exemplo, satélites e aviões, dentre outros. Os componentes redundantes de um satélite permitem que o tempo de operação ininterrupto seja prolongado, aumentando assim, sua vida útil (DUBROVA, 2013).

A implementação de técnicas de tolerância a falhas ou a detecção de falhas necessita de alguma forma de redundância, pois ela pode ter um papel muito importante na confiabilidade de um sistema (AL-ARIAN; GUMUSEL, 1992).

Basicamente, a redundância pode assumir uma destas duas formas:

- Redundância de espaço;
- Redundância de tempo.

Existem ainda algumas outras técnicas de redundância, como a híbrida que usam redundância de espaço e de tempo, e redundância de *software*, no qual é adicionado informações extras, além do que é necessário, para que seja executada uma determinada função (AL-ARIAN; GUMUSEL, 1992).

A redundância de espaço, nada mais é que a replicação física de um *hardware*. Existem duas formas de redundância de espaço: a passiva e a ativa. As técnicas passivas usam o conceito de mascaramento de falhas que ocorrem sem exigir qualquer ação do sistema ou de um operador. Já a redundância ativa requer que uma falha seja detectada antes de ser tolerada. Após a detecção da falha, as ações de localização, contenção e recuperação são realizadas para remover o componente defeituoso do sistema. As técnicas ativas requerem que um sistema seja interrompido e reconfigurado para tolerar falhas (DUBROVA, 2013).

Um exemplo típico de uma redundância de espaço passiva é o sistema de Múltipla Redundância, que pode ser visto na Figura 5. Esse sistema garante que apenas os valores corretos sejam passados para a saída do sistema, apesar da presença de uma falha. Um sistema de Múltipla Redundância funciona da seguinte maneira: uma votação majoritária é usada para determinar o resultado correto. Se um dos módulos falhar, o votador da maioria mascara a falha reconhecendo como correto o resultado dos módulos restantes que permaneceram em funcionamento normal (DUBROVA, 2013).

A redundância de tempo, muitas vezes, é usada na detecção e correção de erros

causados por falhas temporais. Porém, ultimamente tem sido utilizada tanto na detecção quanto na tolerância de erros sendo causados por falhas permanentes ou transitórias no compromisso de diminuir o *hardware* necessário (AL-ARIAN; GUMUSEL, 1992).

Para aplicações de CubeSat, mais precisamente em um OBC, objeto de estudo dessa tese, é necessário um profundo estudo acerca da arquitetura que se deseja conceber, verificando-se a confiabilidade, para que seja identificado onde há necessidade de incluir redundância e de qual tipo, com objetivo de fornecer tolerância a falhas de forma otimizada.

2.3.4 Códigos Corretores de Erros

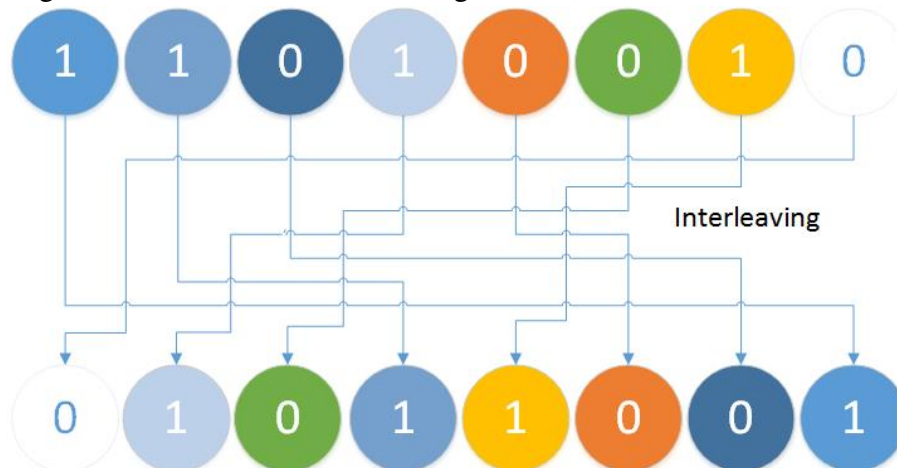
Em sistemas de computação, onde existe comunicação entre os elementos do sistema, como a comunicação entre um processador e uma memória, erros no envio e recepção de dados estão quase sempre presentes, principalmente em ambientes hostil como o espaço. Logo, surgiu a necessidade de haver uma forma para a detecção da existência do erro e a correção do mesmo. Neste cenário, para aumentar a confiabilidade da transmissão, os dados seriam enviados junto com uma informação extra codificando esses dados e decodificando-os para que o receptor receba os dados originais. Esse processo é conhecido como código corretor de erros, estrutura capaz de detectar e corrigir erros em um sistema de comunicação (HAMMING, 1950).

A radiação proveniente do espaço é a causadora dos efeitos SEU que alteram os dados armazenados na memória de um OBC em um nano satélite. Os SEUs predominantes em memórias são o *Single Bit Upset* (SBU) e *Multiple Bit Upset* (MBU). A inversão de um bit ocasionado pelo SEU é chamado de SBU. Já o MBU é fruto do avanço tecnológico nos circuitos integrados. A diminuição da área de uma célula de memória, por exemplo, fez com que uma única partícula injetada sobre a memória atingisse um maior número de células adjacentes ocasionando uma maior incidência de inversão de bit nos dados armazenados na memória (SILVA, 2018).

O *Interleaving* de bits é uma técnica eficaz na mitigação de MBU, pois faz com que várias inversões de bits adjacentes apareçam como vários SBUs em diferentes palavras de dados facilitando o uso de EDACs com correções de um único bit (RADAELLI *et al.*, 2005). O *Interleaving* nada mais é do que a realocação dos dados em diferentes posições de memórias (embaralhamento). Isso faz com que não haja ocorrência de erros múltiplos na mesma palavra armazenada (CUI; ZHANG, 2013). A figura 8 ilustra a técnica do *Interleaving*.

A técnica de *Interleaving* em conjunto com os EDACs, é uma solução bastante utilizada para proteção de memórias em aplicações espaciais como em (RADAELLI *et al.*,

Figura 8 – Processo de Interleaving.



Fonte: O autor.

2005), (CUI; ZHANG, 2013) e (SATOH *et al.*, 2000).

Existem diversos modelos de EDAC para as mais variadas aplicações e com características distintas como custo, potência e área. Nessa tese, serão apresentados alguns EDACs da literatura (Hamming e Hamming Estendido) que foram tomados como base para o desenvolvimento dos códigos projetados no Laboratório de Engenharia de Sistemas de Computação (LESC) (*Column Line Code (CLC)* e *Product Code for Space Applications (PCoSA)*) que também são mencionados nessa tese e, por fim, serão discutidos os EDACs que foram embarcados no Robust OBC (*Double Adjacent Error Correction (DAEC)*, *Triple Adjacent Error Correction (TAEC)* e *Quadruple Adjacent Error Correction (QUAEC)*).

Os códigos de Hamming são códigos baseados na adição de bits de paridade com o objetivo de corrigir erros simples. São capazes de detectar dois bits e corrigir um bit de acordo com o tamanho do código a ser implementado (HAMMING, 1950). Um parâmetro fundamental de um código corretor de erros é a distância mínima. Essa distância entre duas palavras binárias é o número de posições nas quais elas diferem. Em códigos Hamming, que tem como princípio a detecção e correção de erro simples, sua distância mínima é três (SILVA, 2018).

A detecção e correção é feita para todos os bits da palavra código, assim como os bits de paridade. Na decodificação é efetuada a verificação dos bits de paridade da palavra código (conhecida como síndrome) recebida. caso não ocorra nenhum erro, a síndrome é zero. Se ocorrer um erro simples, a síndrome é diferente de zero e o valor indica a posição do bit com erro e para corrigi-lo, basta inverter o seu valor (GAMA, 2008).

A definição do código de Hamming é dada a seguir:

- M é o número de bits de dados e N é o número de bits totais;

– N definida pela equação $N = M + k$, em que k é o número de bits de Hamming.

Essas variáveis se relacionam da seguinte forma:

$$2^k \geq k + M + 1; \quad (2.11)$$

$$2^M \geq \frac{2^N}{N + 1}. \quad (2.12)$$

Já o Hamming Estendido é dado através da adição de um bit extra na distância mínima tornando o valor para quatro. Com essa adição, o código Hamming Estendido é capaz tanto de detectar e corrigir um erro quanto detectar um erro duplo e corrigir um erro simples. Logo, a real diferença entre os códigos Hamming dito anteriormente é a adição de um bit extra de paridade ao final do processo de codificação e no processo de decodificação, a diferença está na verificação da paridade (GAMA, 2008).

O código CLC, desenvolvido por (CASTRO *et al.*, 2016), é baseado nos conceitos de Hamming e paridade e formam um código matricial que corrige múltiplos erros binários. Ele faz uso de uma tabela de correção de erros que indica, a partir da análise das síndromes, o tipo de erro que será corrigido. O CLC possui duas formas de correção: Padrão e Estendido. O modo Padrão faz a correção de erros ao analisar síndromes somente uma vez e o modo Estendido realiza dupla correção que são divididas em duas partes. Com isso, tem-se a verificação de síndromes duas vezes, permitindo a correção de padrões com mais possibilidades de erros. Pelo fato de ser um código com mais processamento, o modo Estendido tem acréscimo de custo na sua utilização (SILVA, 2018).

Para a obtenção da qualificação desta tese, foi desenvolvido um novo código produto baseado em Hamming e paridade tanto nas linhas quanto nas colunas para uso em memórias durante aplicações espaciais, nomeado PCoSA. O PCoSA é um código (64,16), em que uma palavra de 16 bits é codificada em uma de 64. São 16 bits de dados, 12 *checkbits* de linha C1, 7 bits de paridade de linha P1, 21 *checkbits* de coluna C2 e 8 bits de paridade de coluna P2. O PCoSA é apresentado inicialmente com o código Hamming Ham(7,4), porém, tem a taxa de redundância bastante reduzida quando usa o Ham(31,26) (FREITAS *et al.*, 2020).

O código corretor de erros DAEC (GRACIA-MORÁN *et al.*, 2018) pode corrigir um erro em um único bit, ou um erro em dois bits adjacentes, ou pode detectar um erro de 3 bits em rajada ou um erro de rajada de 4 bits. O DAEC precisa de apenas sete bits de código para

uma palavra de dados de 16 bits. Ele possui uma matriz de verificação de paridade H para este código, onde C_i são os bits de código e X_i são os bits de dados primários. Uma vez obtido o H, projeta-se o circuito do codificador/decodificador.

O TAEC (GRACIA-MORÁN *et al.*, 2018) é capaz de corrigir um erro em um único bit, ou um erro em dois bits adjacentes (erros de rajada de 2 bits) ou um erro de rajada de 3 bits, ou pode detectar um erro de rajada de 4 bits. Isso é possível adicionando mais um bit de código. Nesse caso, para uma palavra de dados de 16 bits, o TAEC precisa de oito bits de código. Como no caso do DAEC, possui uma matriz de paridade H onde C_i são os bits de código e X_i são os bits de dados primários. Da mesma forma, a partir de H é consegue-se projetar o circuito codificador/decodificador.

Já o QUAEC (GRACIA-MORÁN *et al.*, 2018) é capaz de corrigir um erro em um único bit, ou um erro em dois bits adjacentes (erros de rajada de 2 bits) ou um erro de rajada de 3 bits ou um erro de rajada de 4 bits. Isso pode ser feito usando apenas nove bits de código. Assim como nos códigos anteriores, C_i são os bits de código e X_i são os bits de dados primários.

Os EDACs são de extrema importância para a proteção de memórias em aplicações espaciais, particularmente as SRAMs, dada a sua eficiência no combate a SEU. Porém, existem outras técnicas, como (NICOLAIDIS *et al.*, 1993) e (VARGAS; NICOLAIDIS, 1994), que são utilizadas na construção de circuitos integrados CMOS e memórias aumentando sua confiabilidade na imunização a TID e SEU.

2.4 Estratégias de Arquiteturas de OBC

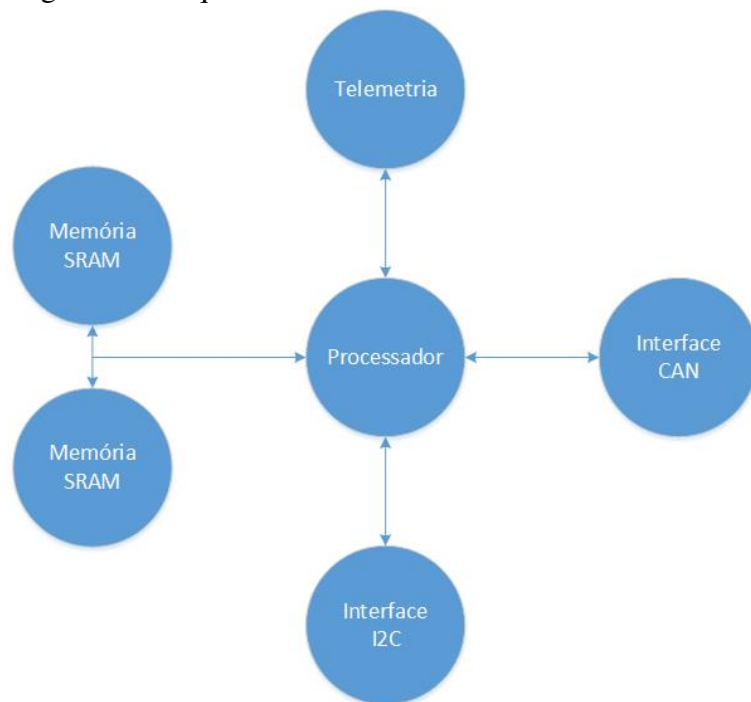
Como dito em (MOTA, 2017), o sucesso do desenvolvimento de um OBC está totalmente vinculado à elaboração de uma arquitetura bem definida à aquilo que ela se propõe a fazer. Uma arquitetura é a representação em blocos das partes do sistema e como elas interagem entre si. Estes blocos são representações de *hardware* ou *software* cujos detalhes de implementação foram abstraídos, deixando apenas as informações de comunicação (MOTA, 2017). A arquitetura de um OBC tende a ser flexível e de uso geral, com o objetivo de aumentar sua capacidade de reutilização (YASHIRO *et al.*, 2001), porém a confiabilidade das arquiteturas serão distintas para cada OBC dado que o mesmo faça uso de componentes Rad-Hard ou use mais ou menos técnicas de tolerância a falhas. De acordo com (WERTZ; LARSON, 1999), um OBC pode ter três diferentes modelos de arquiteturas conforme abaixo:

- Centralizada;

- Distribuída;
- Barramento compartilhado;

Uma arquitetura centralizada possui um elemento central (processador) que se comunica com interfaces ou subsistemas através de nós individuais ou compartilhados. Uma vantagem dessa arquitetura é que uma falha em algum componente ou subsistema não causa perda do sistema. A figura 9 ilustra uma arquitetura centralizada.

Figura 9 – Arquitetura Centralizada.

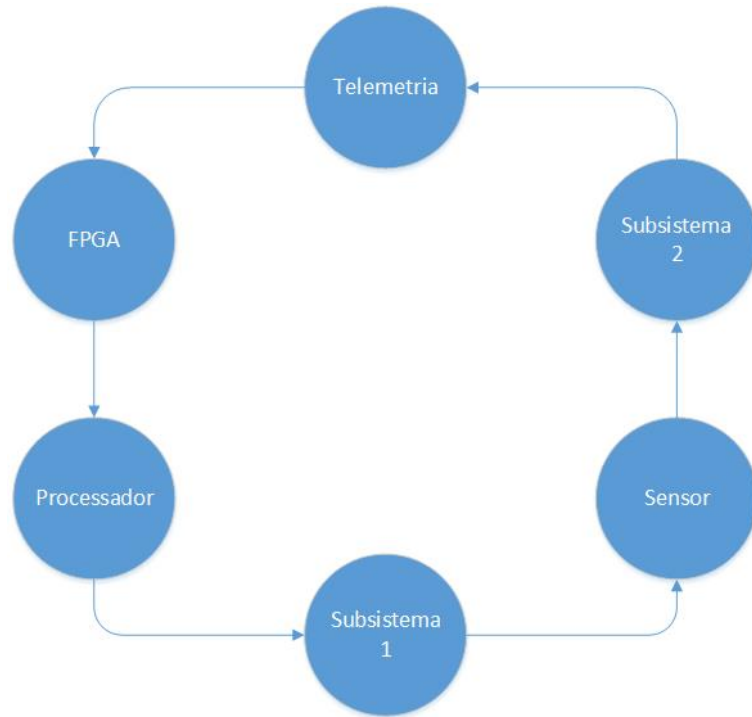


Fonte: O autor.

Na arquitetura distribuída, conforme a figura 10, é estabelecida uma forma de arbitrar o controle de fluxo de informações à medida que os dados são trafegados em um padrão serial. A desvantagem do uso dessa arquitetura é que caso algum componente venha a falhar, o sistema como um todo, também falhará.

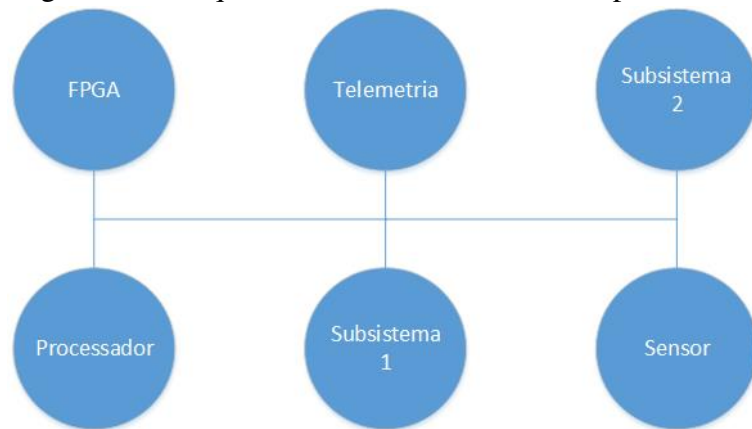
Por último, existe a arquitetura que utiliza um barramento de dados compartilhado para todos os outros componentes e subsistemas. Como a comunicação é feita por endereçamento, a transmissão dos dados é determinística aumentando a confiabilidade na comunicação. Para isso, todos os componentes e subsistemas devem possuir a mesma interface de comunicação. Uma desvantagem dessa abordagem é que caso o barramento compartilhado sofra algum travamento ou curto-circuito, a transmissão de dados será interrompida acarretando em uma falha. A figura 11 ilustra esse tipo de arquitetura.

Figura 10 – Arquitetura Distribuída.



Fonte: O autor.

Figura 11 – Arquitetura com Barramento Compartilhado.



Fonte: O autor.

Uma boa prática em arquitetura de sistemas embarcados para aplicações críticas é na concepção de uma arquitetura híbrida utilizando as vantagens de cada modelo com objetivo de aumentar sua confiabilidade (WERTZ; LARSON, 1999).

O desenvolvimento de computador de bordo pode ser separado em duas categorias. A primeira é fazendo uso de componentes Rad-Hard, no qual eles possam ter um grau de confiabilidade de até 0,9999 e é baseado na prevenção de falhas. A segunda abordagem é em computadores de bordo que possam tolerar algum grau de indisponibilidade ou com confiabilidade mais baixa (como por exemplo, OBCs COTs), possuir um valor em torno de 0,9 e tolerar a ocorrência

ocasional de falhas, sendo protegido por técnicas de mascaramento de falhas que permitem ao sistema emitir resultados corretos mesmo na presença dessas falhas (LIM, 2009).

Para a elaboração da arquitetura de um computador de bordo tolerante a falhas objeto dessa tese, foi pesquisado na literatura, em artigos científicos e no meio corporativo, diversos OBCs COTS com o propósito de analisar e encontrar as lacunas para propor uma contribuição inovadora nessa área de estudo.

Durante o desenvolvimento dessa tese foram utilizados como critérios de seleção de OBCs no estado da arte, para comparação com o Robust OBC, algumas características com objetivo de realizar uma filtragem dada a grande quantidade de OBCs disponíveis no mercado, mas não muito na literatura. A figura 12 ilustra o fluxograma de seleção dos OBCs. Esses critérios foram elaborados utilizando itens de reprovação e itens condicionais que fossem atendidos por pelo menos um deles conforme mostrado a seguir:

1. Não possuir componentes Rad-Hard em sua arquitetura;
2. Seguir o padrão CubeSat;
3. Possuir técnicas de tolerância a falhas como redundância, *watchdog*, circuito de detecção de SEL, EDAC, etc;
4. Possuir componentes críticos com características de tolerância a falhas em sua construção.

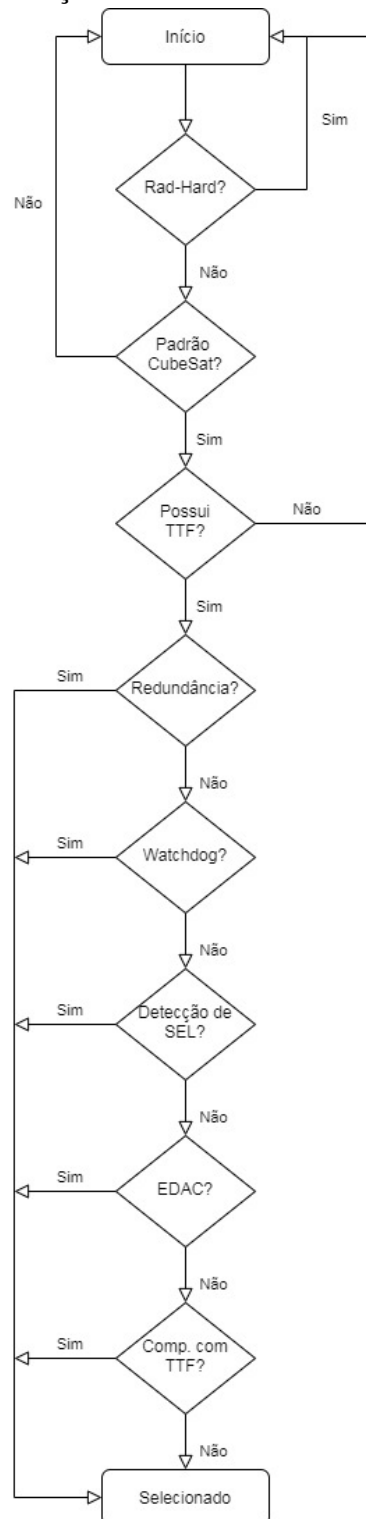
As subseções seguintes fazem um apanhado geral das arquiteturas de alguns computadores de bordo selecionados com o intuito de extrair as técnicas de tolerância a falhas utilizadas e modelar a confiabilidade dessas arquiteturas como forma de adquirir uma métrica para comparação.

2.4.1 OBC 1

O computador de bordo deste artigo (BUSCH; SCHILLING, 2013) possui uma arquitetura tolerante a falhas contra SEU e SEFI na memória interna do processador e SEL, onde pequenos períodos de falhas são aceitos, contanto que o sistema consiga se proteger de danos permanentes, detectar o estado de falha e se recuperar em tempo hábil. A figura 13 ilustra a arquitetura desse OBC.

Para isso, foi implementado um sistema de redundância de processadores na configuração *Master/Slave* que são controlados por um *watchdog*. Em operação normal, o *master* deve enviar um sinal de vida periodicamente ao *watchdog*. Caso o *master* falhe nessa operação, o *watchdog* irá reinicializar os dois processadores e passará o controle do sistema ao *slave*,

Figura 12 – Fluxograma de seleção dos OBCs.

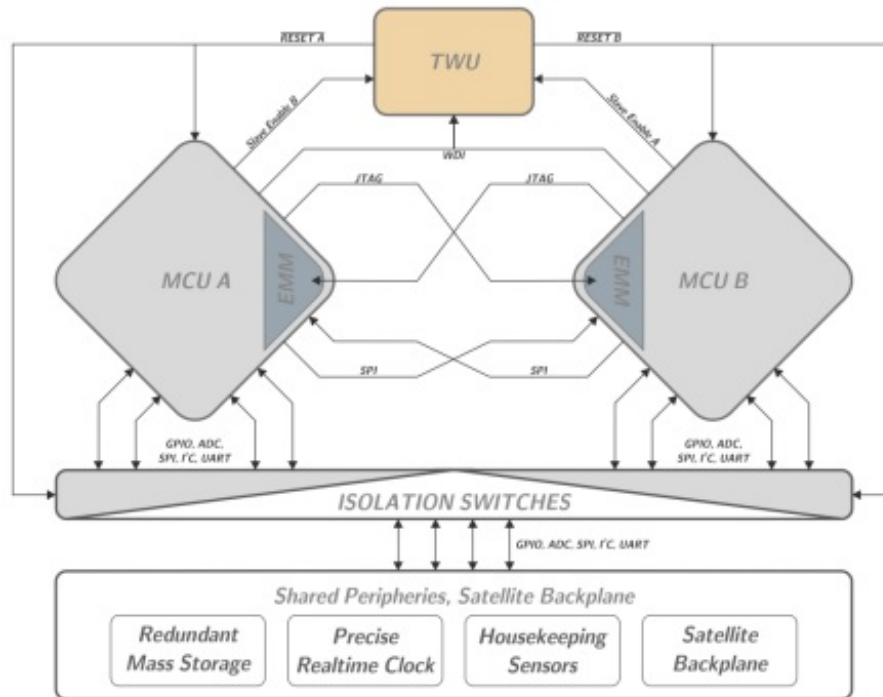


Fonte: O autor.

deixando o antigo *master* em estado de *reset*. Esse sistema é capaz de combater problemas causados por SEU e SEFI.

Já para os casos de SEL, foi implementado um circuito capaz de isolar a alimentação

Figura 13 – Arquitetura do OBC em Busch e Schilling (2013).



Fonte: Busch e Schilling (2013).

dos componentes e, assim, protege-os desse tipo de efeito. Esse circuito é composto por um *watchdog* conectado a uma chave eletrônica. O sistema aguarda por volta de um minuto para realizar uma operação de ciclo de alimentação, caso o processador não dê o sinal periódico no *watchdog*.

Como esse OBC não possui memória SRAM externa, esse artigo não abordou problemas que podem ser solucionados por técnicas de detecção e correção de erros. O único tratamento para isso seria caso um SEU gerasse uma inversão de bit que desse falhas na execução de código do processador, com isso, os processadores são reinicializados e o próximo *master* voltaria a funcionar do início.

No Robust OBC, além de apresentar todas as características de tolerância a falhas contra os efeitos citados acima, ele possui três tipos distintos de EDACs que são selecionados de forma dinâmica de acordo com o tipo de falha transitória que possa vir a ter nas memórias SRAM.

2.4.2 OBC 2

Neste artigo (LEPPINEN *et al.*, 2014), são usados dois processadores TMS570 redundantes a frio do fabricante Texas Instruments. Cada processador possui seus próprios sistemas de *clock* e fontes de alimentação. Outros trabalhos de pesquisa em computadores de

bordo para nano satélites como (MOTA, 2017) e (YUEN; SIMA, 2018), e em OBCs comerciais como (DATA PATTERNS, 2021) utilizaram este processador dada suas características de tolerância a falhas.

Um processador MSP430, respaldado pela organização CubeSat, realiza o papel de árbitro para alternar entre os processadores. Este circuito permite que apenas um processador seja alimentado por vez e que seja habilitado por padrão através de um resistor de *pull-up*, caso o árbitro venha a falhar. Este OBC possui ainda memórias Flash redundantes que são acessadas pelo processador através de chaves coordenadas pelo árbitro do sistema.

Os processadores TMS570 utilizam sinais de erro e de vida conectados ao MSP430. Caso o sinal de vida seja desativado ou um sinal de erro tenha sido enviado, o árbitro fará a troca do processador para que o sistema continue em funcionamento. Certos erros, como erros de comparação entre os *cores* em *lockstep* (este processador possui em sua arquitetura interna dois *cores* redundantes que são defasados no tempo de dois ciclos de *clock*) dentro do TMS570 e erros de memória incorrigíveis, causam a saída de um sinal de erro do pino do módulo de sinalização de erro do processador. Este sinal pode ser usado para reinicializar o sistema ou executar outras ações para responder ao erro.

O funcionamento do sistema acontece quando um processador verifica que está em estado de falha e que ele não pode funcionar, então ele tentará reinicializar, caso não consiga estabelecer a normalidade, o árbitro MSP430 chaveia para o outro processador com base na falta de sinal de vida. A figura 14 ilustra a arquitetura desse OBC.

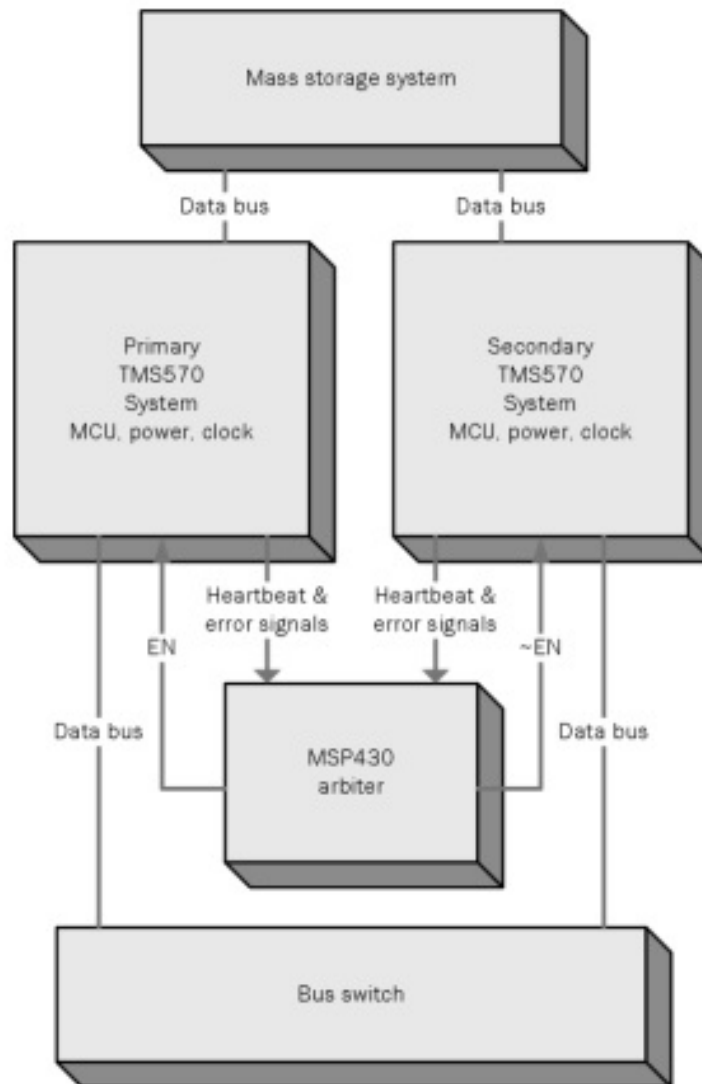
Assim como o OBC 1, essa arquitetura é baseada em redundância de processadores, porém com algumas diferenças como na utilização de um processador para a realização da troca dos processadores redundantes dado que no OBC 1 era implementado utilizando componentes digitais. Outra, e importante, diferença é no uso de processadores com características, em sua construção, de tolerância a falhas.

Uma lacuna observada neste OBC é na ausência de proteção contra SEU nas memórias externas como, por exemplo, no uso de EDACs e assim garantir que possíveis inversões de bits de dados sejam recuperados e não causem prejuízo ao sistema.

2.4.3 OBC 3

O OBC 3 do artigo (BOTMA *et al.*, 2013) possui uma arquitetura tolerante a falhas contra os efeitos provenientes de SEU e SEL. Esses métodos são utilizados para proteção do

Figura 14 – Arquitetura do OBC em Leppinen *et al.* (2014).



Fonte: Leppinen *et al.* (2014).

processador e da memória SRAM externa. No caso do processador, o sistema de proteção contra SEU é verificar e reatribuir frequentemente os valores de todos os registradores importantes. Se um SEU causar uma falha grave, o mesmo método para tolerar SELs pode ser utilizado. Este método foi implementado utilizando um *watchdog* para desligar a alimentação do processador. É necessário um ciclo completo de alimentação para remover um SEL, uma vez que um *reset* normal não é suficiente. Este método é eficiente para lidar com um SEL nos registradores do processador.

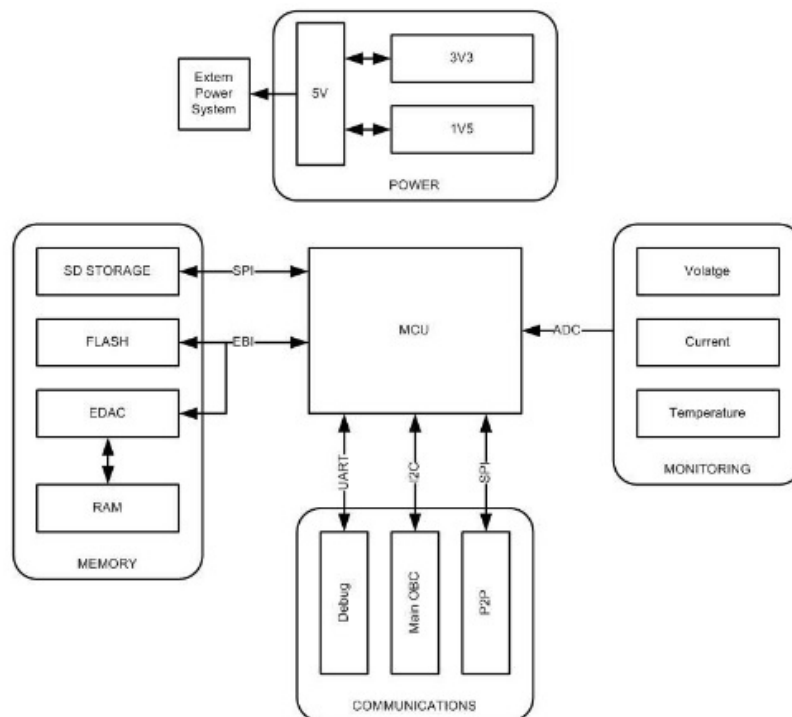
A proteção contra SEL da memória SRAM externa é feita através da leitura de um sensor de corrente que verifica o consumo de corrente da SRAM e alerta o processador caso esse consumo supere o valor predefinido anteriormente. Quando isso acontece, o processador remove a tensão elétrica da memória durante um ciclo de alimentação pré-estabelecido até que o *latchup*

seja totalmente removido.

Para tolerar falhas provenientes dos efeitos SEU, foi utilizado um EDAC baseado em código de bloco linear, do inglês *Linear Block Code* (LBC). Existe um grupo de LBC, chamado de códigos quase-cíclicos binários, que têm a seguinte propriedade $(n, n/2)$. Um código binário quase-cíclico $(16,8)$ não só faria pleno uso do espaço de paridade disponível neste OBC, como também proporcionaria a capacidade de corrigir até dois bits de erro por palavra de código. Além do EDAC, esse sistema possui um *scrubbing* que é feito periodicamente para a memória não acumular erros nos seus dados.

Esse artigo é proveniente da dissertação de mestrado de (BOTMA, 2011) que, em parceria com a empresa CubeSpace, gerou o computador de bordo comercial (CUBE SPACE, 2016). A figura 15 ilustra a arquitetura desse OBC na forma de concepção e a figura 16 ilustra essa mesma arquitetura de maneira comercial.

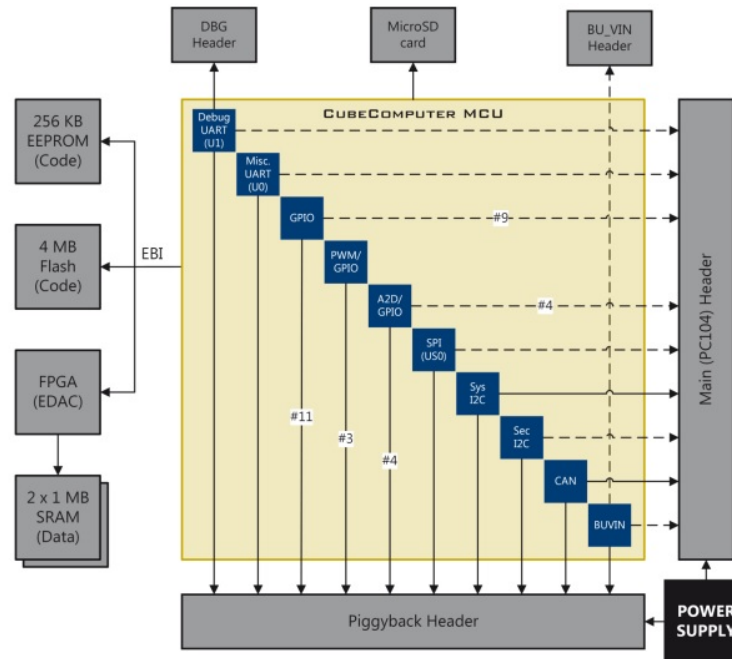
Figura 15 – Arquitetura do OBC em Botma (2011).



Fonte: Botma (2011).

Neste OBC foi verificada algumas lacunas que irão ser exploradas nessa tese, como o fato da ausência de proteção na FPGA que tem um papel importante no provimento do EDAC para a memória SRAM externa. Além disso, mesmo que o processador tenha essas proteções citadas acima, ele é o cérebro do OBC e a confiabilidade sobre ele tende a ser a maior possível, logo se faz necessária algum tipo de redundância para alcançar essa métrica.

Figura 16 – Arquitetura do OBC em Cube Space (2016).



Fonte: Cube Space (2016).

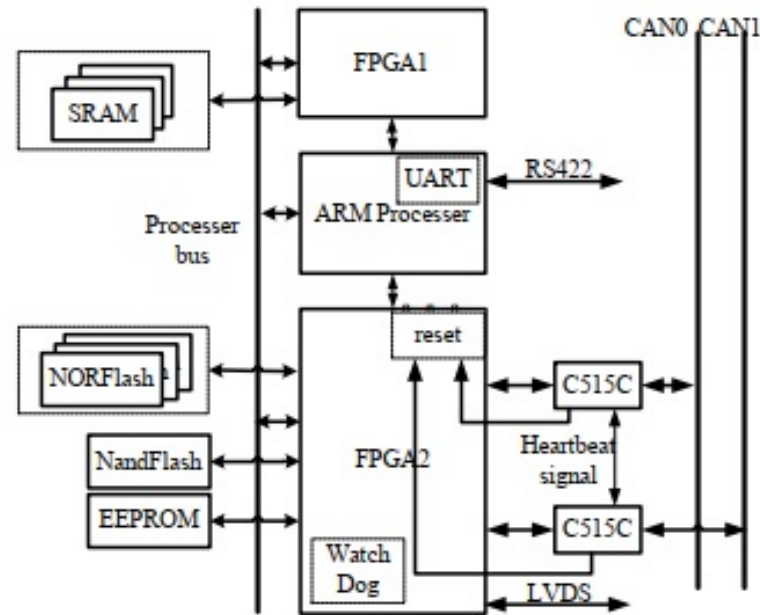
2.4.4 OBC 4

A arquitetura do OBC 4 (TIAN *et al.*, 2012) consiste em um processador ARM, vários tipos diferentes de memória (SRAM, Flash Nor, Flash Nand e EEPROM) para armazenamento de dados e os outros componentes (FPGA e C515C) para provimento de confiabilidade do sistema e implementação de interfaces. A visão geral da arquitetura de *hardware* deste OBC é mostrada na figura 17.

Duas FPGAs *antifuses* se comunicam com o processador ARM por meio do barramento *External Bus Interface* (EBI). A FPGA 1 é conectada a três SRAMs redundantes. A outra FPGA é ligada a três memórias Flash NOR que contém o *bootloader*, a imagem do *kernel* do Linux e a imagem do sistema de arquivos usada para inicializar o sistema. O uso das FPGAs neste OBC é para prover confiabilidade através da técnica de Redundância Modular Tripla, do inglês *Triple Modular Redundancy* (TMR), nas memórias SRAM e Flash NOR.

Dois processadores C515C do fabricante Infineon Technologies são usados para implementar as interfaces CAN principal e redundante, que são as principais interfaces de comunicação no satélite deste artigo. A FPGA 2 fornece a conexão entre o C515C e o processador ARM. Além disso, os C515Cs também são responsáveis por monitorar a tensão, a corrente e a temperatura do sistema para garantir que o mesmo funcione conforme o esperado. Os C515Cs enviam um sinal para o subsistema de alimentação do satélite quando os parâmetros (tensão,

Figura 17 – Arquitetura do OBC em Tian *et al.* (2012).



Fonte: Tian *et al.* (2012).

corrente e temperatura) estão diferentes dos especificados e o OBC seria desligado e, em seguida, religado.

Um *watchdog* é implementado na FPGA 2 para coletar e processar as informações da aplicação em execução no sistema. O *watchdog* é conectado à linha de *reset* do processador ARM, e o processador seria reiniciado pelo *watchdog* assim que uma falha fosse detectada.

Um outro ponto abordado neste artigo é o fato do OBC ser um sistema redundante a frio com dois computadores idênticos, no qual um computador *master* está ligado e um computador *slave* está desligado. Caso haja uma falha catastrófica no OBC *master*, o mesmo seria desligado e o sobressalente seria ligado pelo comando de solo.

Algumas lacunas foram encontradas neste artigo. A primeira é o fato de haver um sistema de monitoramento de tensão e corrente global. Caso haja, por exemplo, um aumento de corrente devido a um SEL em algum componente eletrônico, não há indícios de qual componente foi afetado e com isso a única forma de proteção seria desligando todo o OBC, o que traria perdas na execução desse processo. Um método que poderia amenizar este problema seria na implantação de proteção contra SEL em cada componente crítico do projeto. A segunda seria na inclusão de redundância no processador, dada a sua importância no computador de bordo. Terceiro, o uso de uma FPGA *antifuse* inviabiliza uma das principais características de uma FPGA que é a reconfiguração. Ainda falando da FPGA, a técnica utilizada para proteção das memórias é limitada e de alto custo dada a quantidade de memória necessária para implementar

um TMR. Códigos corretores de erros são mais eficazes e de menor custo para resolver esse problema. E por fim, a forma de troca de OBC através de um sinal proveniente de uma estação terrestre pode não ser tão interessante, pois um travamento ou falha do sistema de telemetria e telecomando do satélite poderá inviabilizar essa troca e o satélite deixará de funcionar.

2.4.5 OBC 5

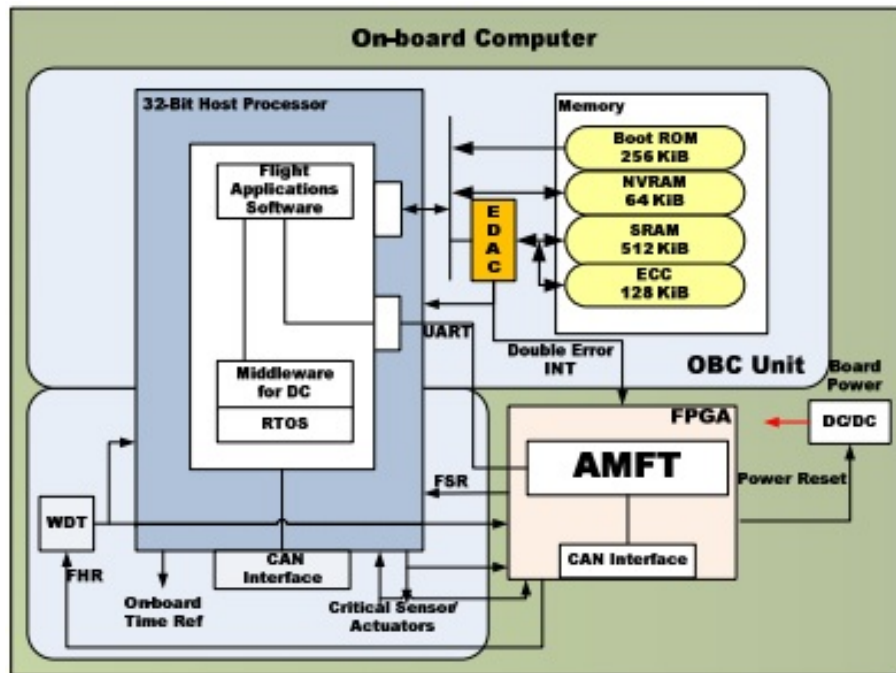
Neste artigo (FAYYAZ *et al.*, 2012), foi desenvolvido um novo computador de bordo distribuído e tolerante a falhas, o qual é composto por várias unidades de OBC, cada uma sendo monitorada por um bloco funcional chamado de *Adaptive Middleware for Fault-Tolerance* (AMFT). O AMFT tem como função monitorar o processador, a memória, o *watchdog* e os periféricos de uma unidade do OBC. Além disso, o AMFT detecta falhas nos OBCs, os isola e reconfigura o OBC. Após a reconfiguração do OBC, o processador do OBC mestre realoca as tarefas do processador com falha para os outros OBCs funcionais do satélite.

Dentro de cada um dos OBCs, há um bloco AMFT. No desenvolvimento do projeto, um número de identificação chamado de OBC ID é atribuído a cada OBC. O menor ID atribuído ao OBC tem a precedência mais alta e é considerado o mestre, enquanto todos os outros são OBCs escravos. Cada computador de bordo executa um algoritmo de adesão. Da mesma forma, os OBC também formam um grupo e apenas o mestre executa a realocação de tarefas em caso de falha. A comunicação entre os OBCs é realizada através da interface CAN. Além disso, existe uma interface de comunicação entre o processador e o AMFT dentro do OBC para atualização de grupo, atualização de estado, detecção de falha, etc.

O AMFT realiza a detecção de falhas, isolamento, recuperação e sincronização da operação de um OBC tolerante a falhas distribuído. Ele é implementado em uma FPGA que se comunica com o processador principal através da UART e sua característica principal é prover proteção contra SEU fazendo uso de TMR em uma arquitetura distribuída, no qual cada OBC realiza uma tarefa predeterminada. Essa arquitetura é escalonável e pode ser aplicada a qualquer número de OBCs. A figura 18 ilustra a arquitetura do OBC e a figura 19 ilustra a conexão entre três OBCs utilizando o sistema AMFT.

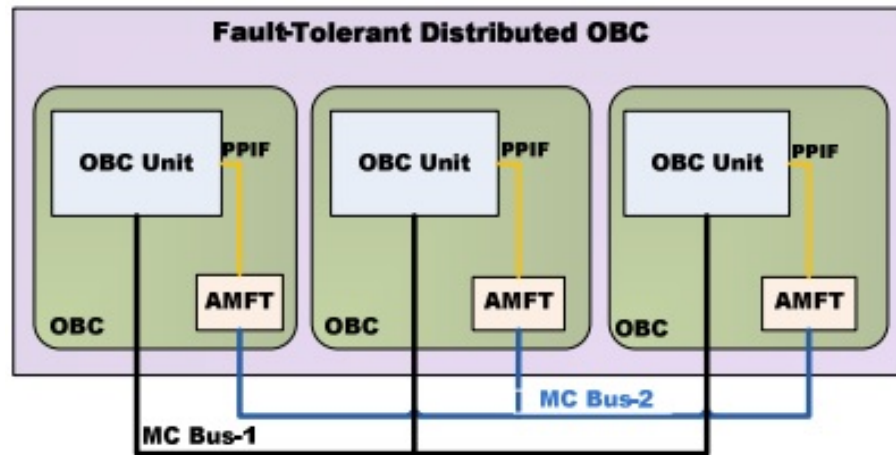
Do ponto de vista de tolerância a falhas, esse sistema não prevê proteção contra SEL e caso haja algum na FPGA, que além de ser peça chave no AMFT e ser um ponto único de falha, não há gerência sobre esse aspecto. Uma alternativa para isso é colocar redundância no sistema de troca de OBCs aumentando sua robustez em algo que é de suma importância no

Figura 18 – Arquitetura do OBC em Fayyaz *et al.* (2012).



Fonte: Fayyaz *et al.* (2012).

Figura 19 – Esquema de conexão do AMFT em Fayyaz *et al.* (2012).



Fonte: Fayyaz *et al.* (2012).

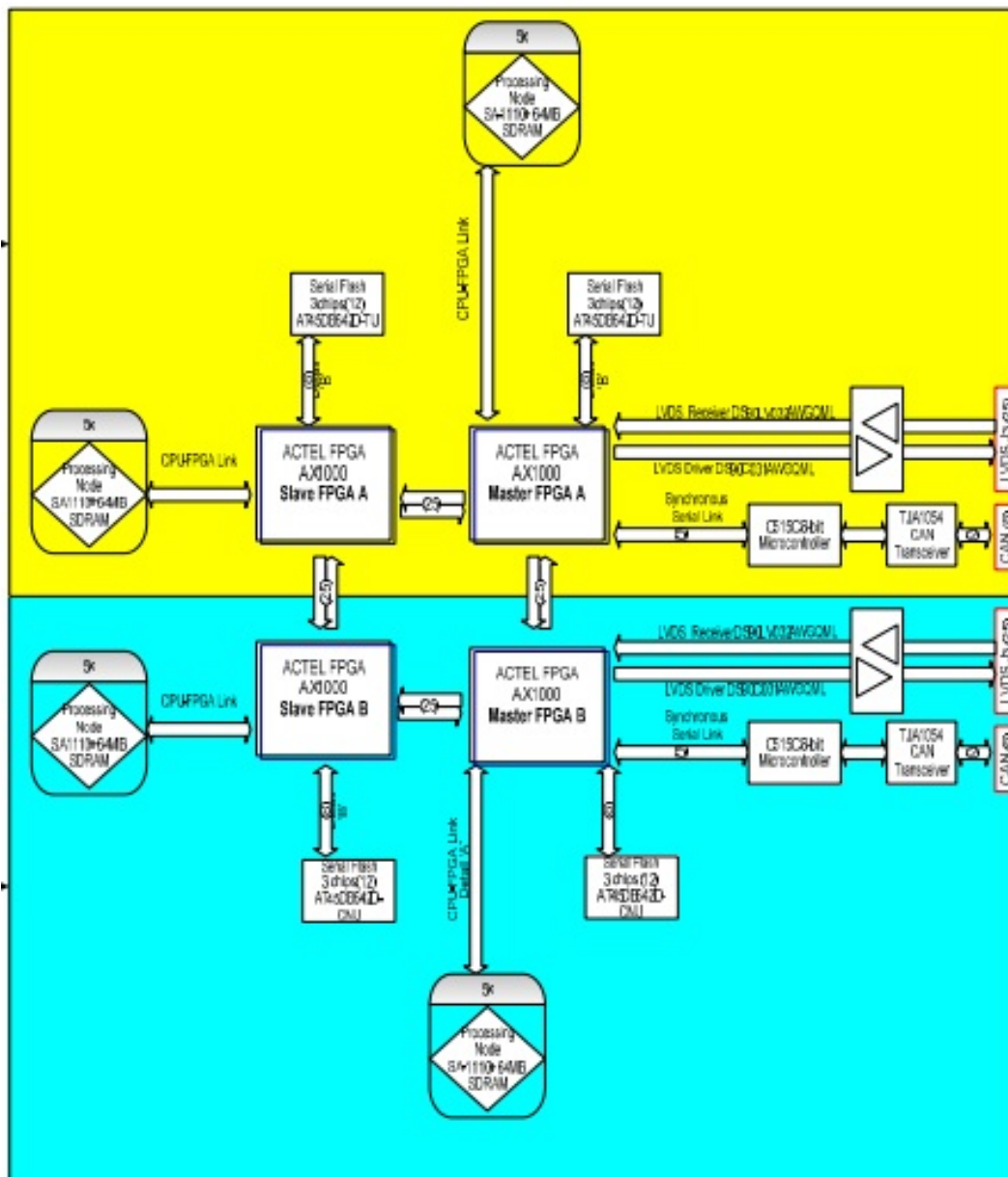
funcionamento correto do AMFT.

2.4.6 OBC 6

O computador de bordo em (LIM, 2009) não utiliza o modelo de nano satélite, porém ele foi incluso nesta análise para a demonstração de que um alto grau de redundância, juntamente de algumas técnicas de tolerância a falhas, faz com que a confiabilidade do sistema que usa componentes COTS chegue a níveis quase idênticos aos OBCs desenvolvidos com componentes Rad-Hard.

A arquitetura é baseada em redundância de *hardware* em que no mesmo OBC é replicado o sistema de computação, ou seja, um OBC possui dois OBCs na mesma placa de circuito impresso. Além disso, em cada sistema de computação existem redundâncias de todos os componentes eletrônicos como duas FPGAs *antifuses*, cinco processadores para se comunicar com cada FPGA, doze memórias Flashes para cada FPGA e, por fim, redundância em todo o sistema de comunicação desse OBC, que por sua vez, já é uma redundância do OBC principal deste satélite que foi desenvolvido utilizando componentes Rad-Hard. A arquitetura pode ser vista na figura 20.

Figura 20 – Arquitetura do OBC em Lim (2009).



Fonte: Lim (2009).

Esta abordagem foge ao conceito sugerido no padrão CubeSat que preza pelo baixo custo, baixo peso, baixa potência e tamanho limitado em 10x10x10 cm, logo sua análise se fez necessária única e exclusivamente para verificação de sua confiabilidade.

2.4.7 OBC 7

Este OBC (GAUSS SRL, 2017) é um OBC comercial e faz uso de uma arquitetura de uso geral, adequada para uma ampla gama de missões de satélite. Ele é projetado para ser flexível e escalável em termos de capacidade de processamento, com o objetivo de manter um consumo de energia muito baixo. É composto por um processador e uma FPGA que atuam em cooperação, fornecendo ao sistema uma redundância de *hardware* e tolerância a falhas de modo comum.

O processador e a FPGA oferecem algumas configurações a serem implementadas (por exemplo, mestre/escravo ou multi-mestre) e a FPGA oferece todas as vantagens da codificação RTL para a implementação de tarefas específicas como controle de atitude e códigos corretores de erro em memória externa ou sistemas genéricos também com IPs de terceiros.

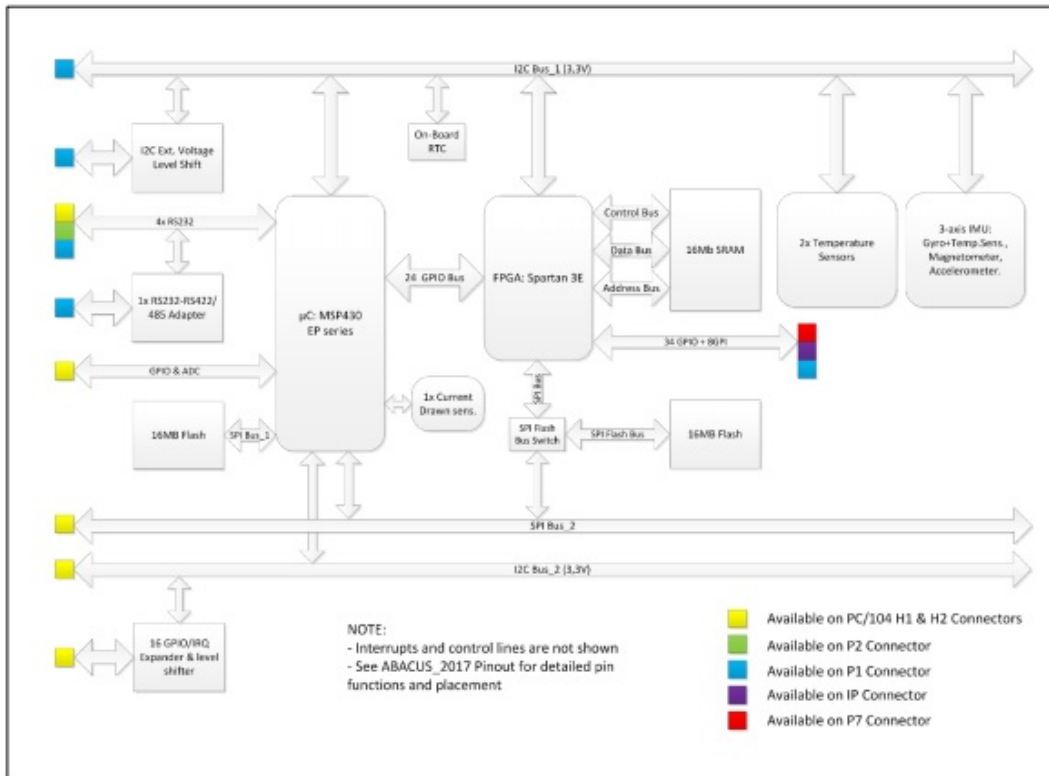
O uso da FPGA nessa arquitetura, fornece a possibilidade de uma alta confiabilidade podendo usar códigos de configuração como o TMR. Além disso, a arquitetura possui vários sensores que fornecem dados de monitoramento de integridade e controle de atitude. Um outro ponto importante nesse OBC é a possibilidade de reconfigurar o código da FPGA e o *firmware* do processador em vôo. A figura 21 ilustra a arquitetura desse OBC.

Diversas são as lacunas encontradas neste OBC. Não há, por exemplo, proteção contra SEL nos componentes críticos, nem circuito de *watchdog* e nem um sistema de redundância de OBC para o caso de o OBC falhar permanentemente.

2.4.8 OBC 8

Outro OBC comercial é o (DATA PATTERNS, 2021) que possui um processador de baixa potência, alto desempenho e várias memórias internas (Flash e SRAM) para atender aos requisitos de processamento de dados das aplicações de nano satélites de padrão CubeSat. Possui várias interfaces de comunicação serial para se comunicar com os outros subsistemas. Ele também possui um conversor analógico/digital para sensores analógicos e sensor de temperatura. O OBC tem uma FPGA para implementação de interfaces seriais personalizadas. Além disso, o OBC tem um *watchdog* externo implementado na FPGA para monitorar continuamente o

Figura 21 – Arquitetura do OBC em Gauss Srl (2017).



Fonte: Gauss Srl (2017).

funcionamento do processador e reiniciá-lo para o caso de haver alguma mudança nas funções redundantes predefinidas com base nos requisitos da aplicação.

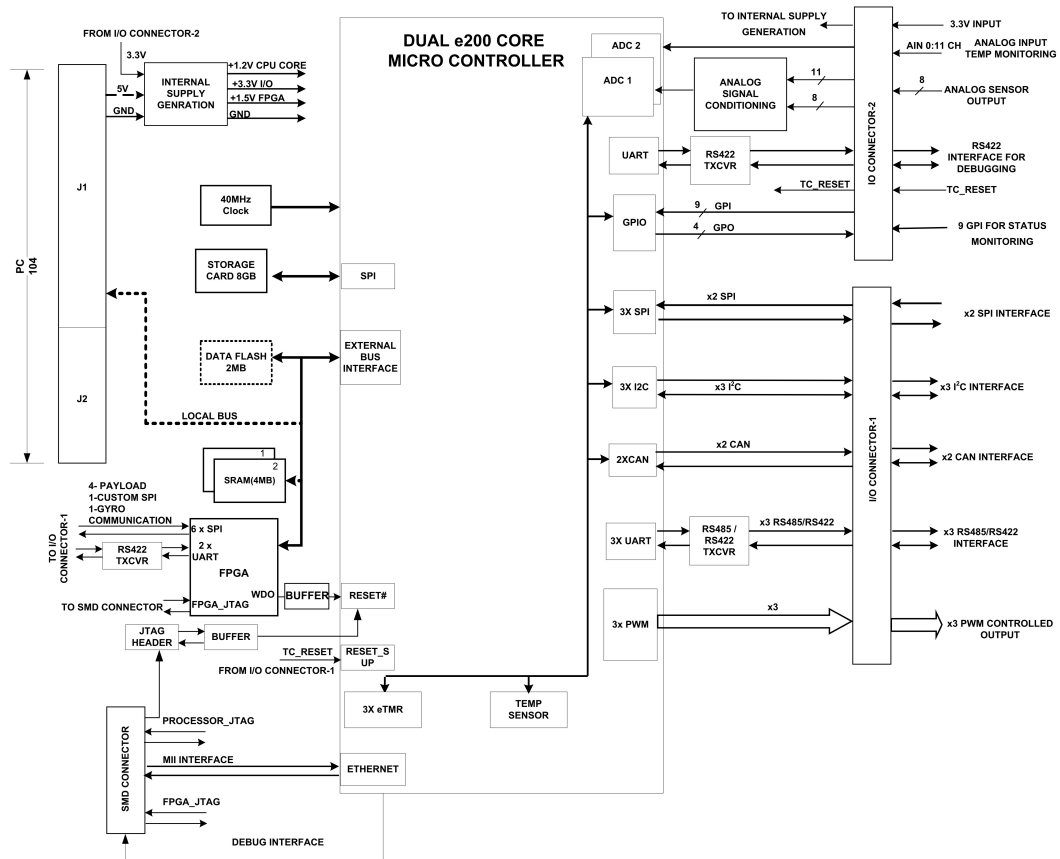
O processador deste OBC é dito tolerante a falhas dado que o mesmo possui dois núcleos que atuam em *lockstep*, códigos corretores de erros em suas memórias internas e diversas outras características de tolerância a falhas. Esse processador é o PXS3020 do fabricante NXP Semiconductors. A figura 22 ilustra a arquitetura desse OBC.

Das técnicas de tolerância a falhas empregadas no OBC 7, a única que o OBC 8 possui a mais é o *watchdog*, além de que é usado um processador com características de tolerância a falhas em sua arquitetura interna. Portanto, esse OBC possui algumas lacunas que podem ser exploradas para a pesquisa e desenvolvimento de um OBC robusto e tolerante a falhas.

2.5 Confiabilidade

Nesta seção será feita uma modelagem das arquiteturas dos OBCs apresentados com o intuito de extrair a confiabilidade de cada OBC para poder propor uma solução robusta com base na análise das técnicas de tolerância a falhas empregadas nos OBCs em conjunto com as métricas obtidas na verificação das confiabilidades.

Figura 22 – Arquitetura do OBC em Data Patterns (2021).



Fonte: Data Patterns (2021).

Esta modelagem é feita através da análise somente do sistema de computação, ou seja, não serão considerados interfaces, fontes de alimentação, etc. A análise ficará por conta somente de processadores, FPGAs e memórias.

Foi verificado que existem padrões nas arquiteturas dos sistemas de computação dos OBCs pesquisados. Pôde-se separar as arquiteturas desses OBCs em quatro padrões dado que para fins de cálculo de confiabilidade, as expressões matemáticas serão as mesmas.

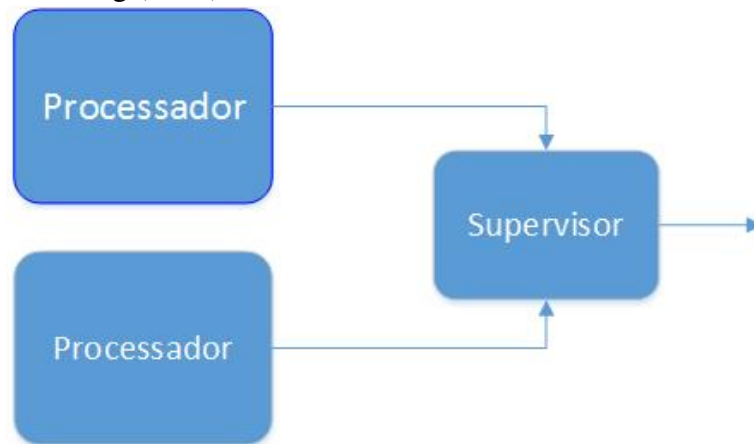
Nesse contexto, a confiabilidade será analisada de duas formas: através de falhas permanentes e falhas transitórias. No modelo de falhas permanentes será analisado na forma de diagramas de blocos, já no modelo de falhas transitórias a análise será feita através do uso do processo de Markov pelo fato desse processo possuir a condição de recuperação de falhas, algo típico de técnicas de tolerância a falhas contra fenômenos como SEU (FAYYAZ; VLADIMIROVA, 2014).

Para o caso de redundância de OBCs, a análise será feita baseada somente em falhas permanente, pois a troca de OBC só ocorrerá caso haja uma falha catastrófico a ponto de invalidar o uso do OBC em uso.

2.5.1 OBC 1

A arquitetura do OBC 1 pertence ao padrão 1 e é do tipo redundância de espaço ativa a frio, dado que somente o processador mestre está operando e o escravo está desligado. A troca de processador é feita por um supervisor e acontece quando o mestre falha. O modelo gerado através das informações obtidas em (BUSCH; SCHILLING, 2013) é visto na figura 23.

Figura 23 – Modelagem da arquitetura do OBC em Busch e Schilling (2013).



Fonte: O autor.

Após a modelagem do OBC, são extraídas as equações de confiabilidade para falhas permanentes e transitórias.

– Confiabilidade para falhas permanentes:

De acordo com (FAYYAZ; VLADIMIROVA, 2014), a expressão matemática que define esse modelo de arquitetura para falhas permanentes é

$$R_{obc1}(t) = R_{sup} * (2e^{-\lambda_{pro}t} - e^{-2\lambda_{pro}t}). \quad (2.13)$$

Onde:

R_{obc1} é a confiabilidade do OBC 1;

R_{sup} é a confiabilidade do supervisor;

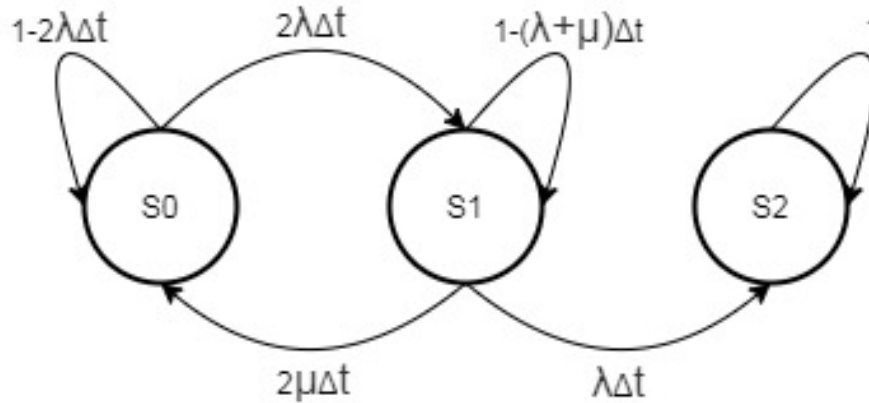
λ_{pro} é a taxa de falha do processador.

– Confiabilidade para falhas transitórias:

Para o levantamento da expressão matemática da confiabilidade da arquitetura para falhas transitórias foi usado um modelo de Markov com dois processadores idênticos. Inicialmente, ambos os processadores estão no estado $S0$. Em caso de falha, a transição do estado $S0$ para $S1$ é ativada e as tarefas em execução no processador com falha são migradas para o

processador íntegro. No estado $S1$, um dos processadores está trabalhando e compartilhando a carga de trabalho das tarefas do processador com falha (FAYYAZ; VLADIMIROVA, 2014). A figura 24 ilustra o modelo de Markov desse OBC para falhas transitórias.

Figura 24 – Modelo de Markov para o OBC em Busch e Schilling (2013).



Fonte: O autor.

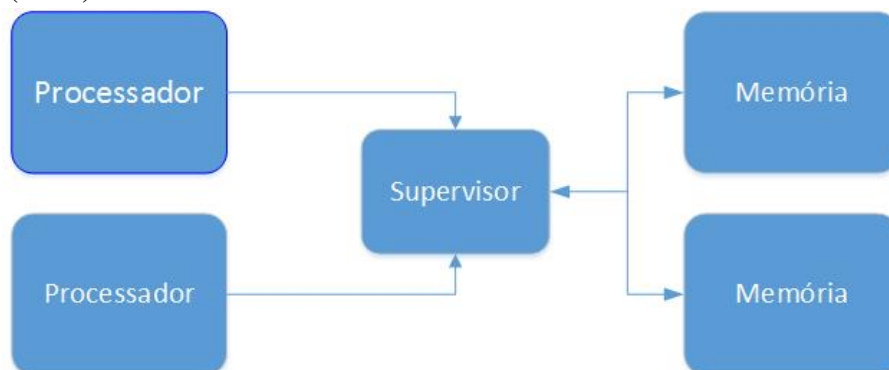
Ainda de acordo com (FAYYAZ; VLADIMIROVA, 2014), a expressão matemática que define o modelo dessa arquitetura para falhas transitórias é

$$R_{obc1}(t) = \left(2 + \frac{\mu}{\lambda}\right)e^{-\lambda t} - \left(1 + \frac{\mu}{\lambda}\right)e^{-2\lambda t}. \quad (2.14)$$

2.5.2 OBC 2

No caso do OBC 2, que pertence ao padrão 2, sua arquitetura é parecida ao OBC 1. A diferença está no tipo do supervisor e no acréscimo de memórias externas redundantes. Logo o modelo gerado para o cálculo de confiabilidade deste OBC é visto na figura 25.

Figura 25 – Modelagem da arquitetura do OBC em Leppinen *et al.* (2014).



Fonte: O autor.

A modelagem desse OBC foi feita separando a arquitetura em dois grupos: um

grupo contendo os dois processadores e o supervisor (*RI*) e o outro grupo contendo as memórias redundantes (*R2*). Logo a equação que representa esse OBC é:

$$R_{obc2}(t) = R1xR2. \quad (2.15)$$

– Confiabilidade para falhas permanentes:

Como o grupo *RI* é igual ao OBC 1, logo sua expressão matemática é igual a Equação 2.13. Já o grupo *R2* é proveniente da Equação 2.7, logo a expressão final da confiabilidade do OBC 2 para falhas permanentes é:

$$R_{obc2}(t) = R_{sup}*(2e^{-\lambda_{pro}t} - e^{-2\lambda_{pro}t})(1 - (1 - e^{-\lambda_{mem}t})(1 - e^{-\lambda_{mem}t})). \quad (2.16)$$

Onde:

R_{obc2} é a confiabilidade do OBC 2;

R_{sup} é a confiabilidade do supervisor;

λ_{pro} é a taxa de falha do processador;

λ_{mem} é a taxa de falha da memória.

– Confiabilidade para falhas transitórias:

Como não há, nessa arquitetura, recuperação de falhas transitórias nas memórias, não haverá um modelo de Markov para *R2*. A mesma usará o modelo de confiabilidade de diagrama de blocos conforme a equação 2.7. Dado que o grupo *RI* desse OBC é igual ao OBC 1, logo seu modelo de Markov é igual a figura 25 e sua expressão matemática é igual a Equação 2.14. Com isso, a equação final para falhas transitórias do OBC 2 é:

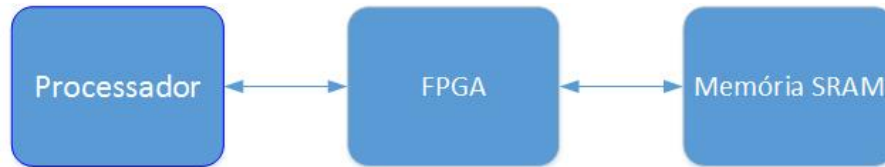
$$R_{obc2}(t) = [(2 + \frac{\mu}{\lambda})e^{-\lambda t} - (1 + \frac{\mu}{\lambda})e^{-2\lambda t}](1 - (1 - e^{-\lambda_{mem}t})(1 - e^{-\lambda_{mem}t})). \quad (2.17)$$

2.5.3 OBC 3

Este OBC, que pertence ao padrão 3, adotou uma arquitetura do tipo serial em que a comunicação entre o processador e memória SRAM é feita através da FPGA, onde é realizada a detecção e correção de erros de uma possível inversão de bits proveniente de um SEU. Este modelo de arquitetura é dependente da confiabilidade de cada componente e, como cada um é um SPF, qualquer falha permanente em algum deles inviabilizaria o satélite. A figura 26 ilustra o modelo gerado do OBC 3.

– Confiabilidade para falhas permanentes:

Figura 26 – Modelagem da arquitetura do OBC em Botma *et al.* (2013).



Fonte: O autor.

Como essa arquitetura é serial, sua expressão matemática é dada de acordo com a Equação 2.6. Então, sua confiabilidade é vista a seguir:

$$R_{obc3}(t) = (e^{-\lambda_{pro}t})(e^{-\lambda_{fpga}t})(e^{-\lambda_{mem}t}). \quad (2.18)$$

Onde:

R_{obc3} é a confiabilidade do OBC 3;

λ_{pro} é a taxa de falha do processador;

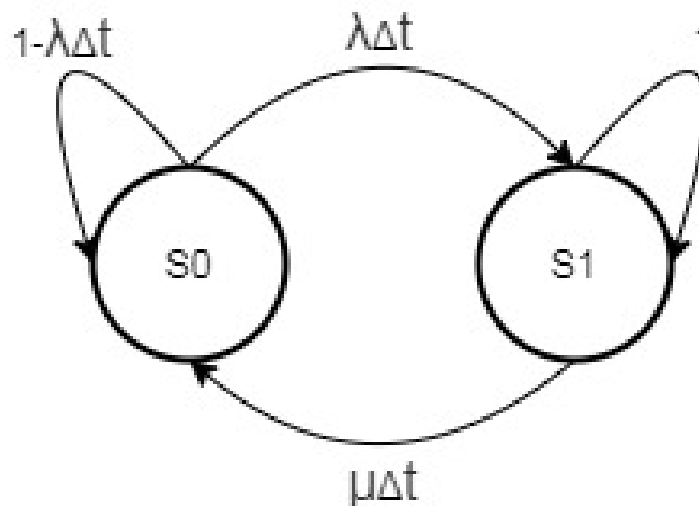
λ_{fpga} é a taxa de falha da FPGA;

λ_{mem} é a taxa de falha da memória.

– Confiabilidade para falhas transitórias:

Nessa análise, a FPGA é o único componente eletrônico da arquitetura que é capaz de se recuperar/reconfigurar contra uma inversão de bits provenientes de um SEU, logo somente ela usará o modelo de Markov para a aquisição da confiabilidade para falhas transitórias. Conforme (FAYYAZ; VLADIMIROVA, 2014), a análise de um componente único corresponde ao modelo centralizado e possui o modelo de Markov ilustrado na figura 27.

Figura 27 – Modelo de Markov para o OBC em Botma *et al.* (2013).



Fonte: O autor.

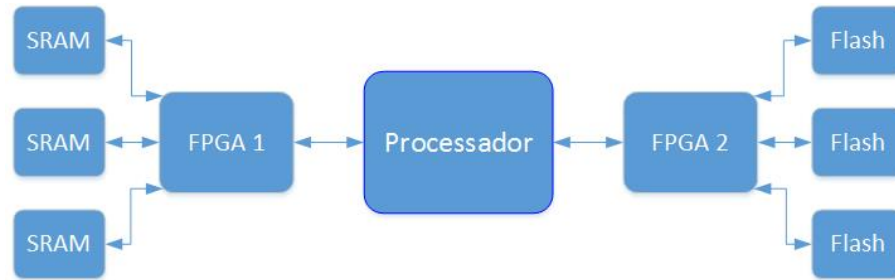
Com isso, a expressão matemática para falhas transitórias é:

$$R_{obc3}(t) = (e^{-\lambda_{prot}t}) \left[\frac{1}{(\lambda_{fpga} + \mu)} (\mu + \lambda_{fpga} e^{-(\lambda_{fpga} + \mu)t}) \right] (e^{-\lambda_{mem}t}). \quad (2.19)$$

2.5.4 OBC 4

O OBC 4, que pertence ao padrão 4, adota uma arquitetura com mais elementos que integram o sistema de computação. Entretanto, é utilizada redundância somente nas memórias fazendo com que os demais componentes atuem como um SPF. As FPGAs tem um papel de votador nessa arquitetura. Além disso, esse sistema possui uma redundância de OBC no modo frio que é trocado, em caso de falha, através de um comando terrestre, em outras palavras o supervisor é uma estação na terra. A figura 28 ilustra o modelo gerado do OBC 4.

Figura 28 – Modelagem da arquitetura do OBC em Tian *et al.* (2012).



Fonte: O autor.

– Confiabilidade para falhas permanentes:

Para a obtenção da expressão matemática da confiabilidade da arquitetura do OBC 4 para falhas permanentes será necessário o desmembramento do sistema em três regiões. A região *R1* será composta pela FPGA 1 e as memórias SRAM, a região *R2* será somente o processador e a região *R3* é composta pela FPGA 2 e as memórias Flash. As regiões *R1* e *R2* foram implementadas através de um TMR e de acordo com (FAYYAZ; VLADIMIROVA, 2014), a expressão matemática dessa configuração é:

$$R_{tmr}(t) = R_{vot} * (3e^{-2\lambda_{mem}t} - 2e^{-3\lambda_{mem}t}). \quad (2.20)$$

Onde:

R_{tmr} é a confiabilidade do TMR;

R_{vot} é a confiabilidade do votador (FPGA);

λ_{mem} é a taxa de falha da memória.

Logo, a confiabilidade da arquitetura para falhas permanentes do OBC 4 é:

$$R_{obc4}(t) = (R_{tmr1})(e^{-\lambda_{pro}t})(R_{tmr3}). \quad (2.21)$$

Onde:

R_{tmr1} é a confiabilidade do TMR da região 1;

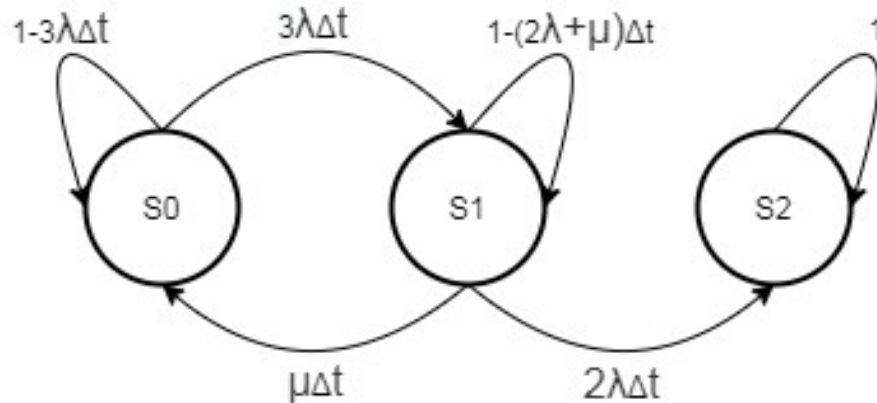
λ_{pro} é a confiabilidade do processador;

R_{tmr3} é a confiabilidade do TMR da região 3.

– Confiabilidade para falhas transitórias:

Como dito anteriormente, as FPGAs fazem um papel de votador dentro dessa arquitetura, logo há a necessidade da aplicação do modelo de Markov nesse sistema dada sua característica de tolerância a falhas contra SEUs. De acordo com (FAYYAZ; VLADIMIROVA, 2014), o modelo de Markov para o TMR é ilustrado na figura 29.

Figura 29 – Modelo de Markov para o OBC em Tian *et al.* (2012).



Fonte: O autor.

Nesse cenário, cada memória tem a mesma taxa de falha λ e taxa de recuperação μ . Inicialmente, todas as memórias estão no estado $S0$ e a probabilidade de falha na transição do estado $S0$ para $S1$ é dada por 3λ . Assim, a expressão matemática desse modelo de Markov é:

$$R_{tmr}(t) = \left(3 + \frac{\mu}{\lambda}\right)e^{-2\lambda t} - \left(2 + \frac{\mu}{\lambda}\right)e^{-3\lambda t}. \quad (2.22)$$

Com isso, o modelo de arquitetura do OBC 4 é baseada na Equação 2.6 e a expressão matemática final de sua confiabilidade é:

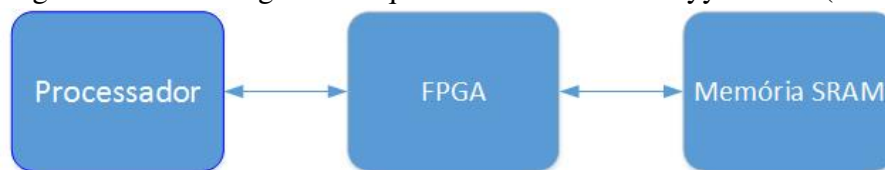
$$R_{obc4}(t) = (R_{tmr1})(e^{-\lambda_{pro}t})(R_{tmr3}). \quad (2.23)$$

Como este OBC possui uma proposta de redundância de OBCs a frio conforme o modelo da figura 23 a expressão matemática para falha permanente é igual à Equação 2.13.

2.5.5 OBC 5

Assim como o OBC 3, a arquitetura desse OBC, que pertence ao padrão 3, é serial e funciona da mesma forma no que diz respeito ao sistema de computação. Sua diferença é que ele possui um sistema de supervisão de redundância de diversos OBCs. Com isso, será calculada a confiabilidade de uma unidade desse OBC e em seguida será calculada a confiabilidade do sistema com três OBCs dando uma métrica ao sistema como um todo. A figura 30 ilustra o modelo gerado do OBC 5.

Figura 30 – Modelagem da arquitetura do OBC em Fayyaz *et al.* (2012).



Fonte: O autor.

A análise da confiabilidade de somente uma unidade desse OBC é igual ao do OBC 3, portanto, suas expressões matemáticas de confiabilidade para falhas permanentes e transitórias são iguais as Equações 2.18 e 2.19 respectivamente.

Como, no caso desse OBC, sua arquitetura de redundância de OBC é do tipo distribuída e não há evidências em (FAYYAZ; VLADIMIROVA, 2014) da análise do mesmo para falhas permanentes, esse sistema será analisado de forma TMR para o caso citado.

– Confiabilidade para falhas permanentes:

Conforme a região $R1$ e $R3$ do OBC 4, a expressão matemática que configura o TMR desse OBC é dada pela Equação 2.20.

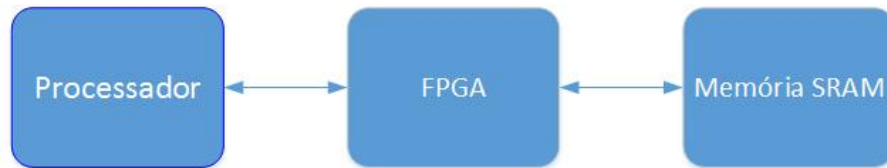
2.5.6 OBC 6

Como dito na subseção 2.4.6, este OBC não possui o padrão CubeSat. A sua análise foi feita para a demonstração de que com um projeto que englobe as mais variadas técnicas de tolerância a falhas juntamente com o fato de não haver restrição de espaço e custo para este tipo de satélite, diferente dos nano satélites que fazem uso do padrão CubeSat, uma alta confiabilidade é facilmente alcançada. Com isso, (LIM, 2009) demonstrou-se que a confiabilidade dessa arquitetura é de 0,9753 para o caso de sobrevivência desse OBC, que é quase igual (0,976) a confiabilidade da arquitetura do OBC principal deste satélite que foi projetado usando os componentes eletrônicos Rad-Hard.

2.5.7 OBC 7

Assim como o OBC 3 e 5, o OBC 7 pertence ao padrão 3 e também possui uma arquitetura serial entre o processador, a FPGA e memória SRAM. Essa técnica é bastante utilizada tanto em OBCs comerciais quanto em OBCs da literatura fruto de pesquisa científicas dada sua importante contribuição na área de tolerância a falhas para este tipo de aplicação. Logo, seu modelo é visto na figura 31.

Figura 31 – Modelagem da arquitetura do OBC em Gauss Srl (2017).



Fonte: O autor.

Com isso, as expressões matemáticas para o cálculo da confiabilidade tanto para falhas permanentes como para falhas transitórias são mostradas nas Equações 2.18 e 2.19 respectivamente.

2.5.8 OBC 8

O OBC 8 pertence ao padrão 3 e tem a mesma arquitetura do OBC 3, 5 e 7. Vale ressaltar a importância desse tipo de arquitetura em OBCs que fazem uso do padrão CubeSat. Esse modelo é o mais encontrado quando se deseja proteção contra SEU em memória SRAM externa, dado que é possível a implementação em FPGA de EDACs para a solução desse problema.

Como diferença dos outros OBCs, este OBC possui a possibilidade de aplicação de redundância de OBC e isso será levada em consideração no cálculo de confiabilidade. Assim como os OBCs citados, seu modelo de arquitetura é serial e pode ser visto na figura 32.

Figura 32 – Modelagem da arquitetura do OBC em Data Patterns (2021).



Fonte: O autor.

Para a análise da confiabilidade de somente um OBC, suas expressões matemáticas

para falhas transitórias permanentes e transitórias seguem as Equações 2.18 e 2.19 respectivamente e para a análise de confiabilidade de redundância a frio do OBC 8, a expressão matemática para falhas permanentes é equivalente a equação 2.13.

2.6 Conclusão

Os OBCs escolhidos para comparação com o Robust OBC passaram por um processo de seleção bastante criterioso com objetivo de afunilar dada a grande quantidade de OBCs disponível, principalmente no meio corporativo. Logo, a falta de informação desses OBCs provenientes de segredo industrial inviabilizou uma maior seleção desse nicho de OBC. Olhando para OBCs frutos de pesquisas científicas, a quantidade diminui bastante, porém as informações são detalhadas e, conseqüentemente, sua análise se torna mais elaborada. Com isso, foi criada uma tabela contendo um resumo da seleção dos OBCs que pode ser vista na Tabela 1.

Tabela 1 – Tabela com os OBCs selecionados

OBC	Padrão	Rad-Hard?	Técnica de tolerância a falhas?	Qual técnica?	Componente com técnica de tolerância a falhas?
OBC 1	1	Não	Sim	Redundância, <i>Watchdog</i> e Circuito de detecção de SEL	Não
OBC 2	2	Não	Sim	Redundância e <i>Watchdog</i>	Sim
OBC 3	3	Não	Sim	Redundância, <i>Watchdog</i> , Circuito de detecção de SEL e EDAC	Não
OBC 4	4	Não	Sim	Redundância e <i>Watchdog</i>	Não
OBC 5	3	Não	Sim	<i>Watchdog</i> e EDAC	Não
OBC 7	3	Não	Sim	EDAC	Não
OBC 8	3	Não	Sim	<i>Watchdog</i> e EDAC	Sim

Fonte: O autor.

Visto que existem técnicas comuns nos OBCs selecionados, foi verificado que há semelhanças nas arquiteturas dos mesmos nos quais há variações apenas nas inovações que cada um dos computadores de bordo propõem como contribuição, detalhes que trazem benefícios no uso dos OBCs em suas respectivas missões.

Depois de analisar cada modelo de arquitetura dos OBCs, foi verificado que as falhas permanentes são dependentes das taxas de falhas dos componentes que compõem a arquitetura e que as falhas transitórias são tanto dependentes das taxas de falhas quanto das taxas de recuperação. De acordo com (LIM, 2009) os valores dessas taxas são de difícil acesso no qual somente alguns fabricantes de componentes eletrônicos disponibilizam e algumas entidades oferecem esses valores através de testes realizados e armazenados em banco de dados para a

comunidade. Outra dificuldade encontrada foi da utilização de um *software* para obtenção da confiabilidade das arquiteturas, tendo em vista que esses programas computacionais são pagos e de alto custo, inviabilizando seu uso por parte da Universidade Federal do Ceará.

No próximo capítulo será feito todo o desenvolvimento do Robust OBC com base na pesquisa realizada e nas melhorias encontradas nas lacunas dos OBCs selecionados. Em seguida, será feito o estudo de confiabilidade do Robust OBC para poder comparar com os demais OBCs visto no capítulo 2.

3 ROBUST OBC: UM SISTEMA DE COMPUTAÇÃO DE BORDO TOLERANTE A FALHAS DE ALTA CONFIABILIDADE

A necessidade de uma arquitetura de um OBC no padrão CubeSat (THE CUBESAT STANDARD, 2021) que atenda as lacunas encontradas na literatura surgiu pelo fato de haver uma demanda cada vez maior em computadores de bordo, que fazem uso de componentes COTS, que sejam capazes de satisfazer a relação custo/benefício existente no mercado de nano satélites. Dada a alta confiabilidade que se propõe com essa arquitetura, este OBC será capaz de ser usado nas mais variadas missões, sendo elas triviais ou críticas.

Com isso, este capítulo descreve todas as estratégias utilizadas na concepção da arquitetura tolerante a falhas, no cálculo de sua confiabilidade para comparação com as arquiteturas vistas no capítulo 2 e na implementação das técnicas de tolerância a falhas dos blocos existentes na arquitetura.

3.1 Visão geral da arquitetura

Conforme visto no capítulo 1, os efeitos que causam maiores problemas para os nano satélites em órbitas baixas são TID, SEU e SEL (GEORGE; WILSON, 2018). Diversas são as técnicas usadas para combater os problemas causados por esses efeitos. Nesse trabalho, o foco da solução contra esses efeitos será no SEU e SEL, dado que para o TID somente os componentes Rad-Hard são realmente eficiente contra esse tipo de efeito, mas como a missão de um CubeSat gira em torno de 2 anos, logo o acúmulo de radiação nas camadas isolantes dos componentes COTS não é uma preocupação para essa aplicação.

Como apresentado, no capítulo 2, a necessidade de proteção de memória SRAM contra inversão de bits provenientes de SEU. Dessa forma, uma FPGA, também descrita no capítulo 2, se fez necessária na arquitetura proposta para a implementação de EDACs e, assim, aumentar a confiabilidade dos dados transmitidos pelo processador e escritos nas memórias. No Robust OBC haverá duas memórias SRAM e duas memórias Flash conectadas à FPGA, podendo ser configuradas como redundantes ou extensíveis conforme a aplicação desejada.

Como o processador é o elemento principal da arquitetura, sua confiabilidade deve ser a maior possível. Além do processador selecionado ter características de tolerância a falhas em sua construção conforme (MOTA, 2017), (LEPPINEN *et al.*, 2014) e (YUEN; SIMA, 2018), será utilizado também um sistema de redundância a frio desse componente para uma maior segurança dos mesmos contra SEL e SEU. Em resumo, haverá uma dupla proteção em cada processador

dado que o mesmo possui internamente defesa contra sub e sobre tensão de alimentação (SEL) e possui um sinal de erro causado por falhas em registradores internos, além de possuir códigos corretores de erros nas memórias internas (SEU).

O sistema de troca de processadores é feito pela FPGA, que irá monitorar eventuais falhas transitórias e permanentes e caso haja detecção de alguma dessas falhas, a FPGA desligará um processador e ligará o outro dando-o todo o controle do OBC para o mesmo.

Uma outra e importante funcionalidade foi atribuída a arquitetura do Robust OBC. Um sistema modular com até três OBCs foi incorporado com o propósito de aumentar a confiabilidade do sistema de computação do nano satélite. Um OBC é inicialmente ligado e em estado de operação e os outros dois permanecem desligados e só serão ativados caso haja uma falha catastrófica no primeiro OBC. Se isso ocorrer, todo o controle do nano satélite será passado para o segundo OBC e em seguida para o terceiro. Essa numeração dos OBCs é feita em *hardware* e em tempo de implantação do mesmo no satélite.

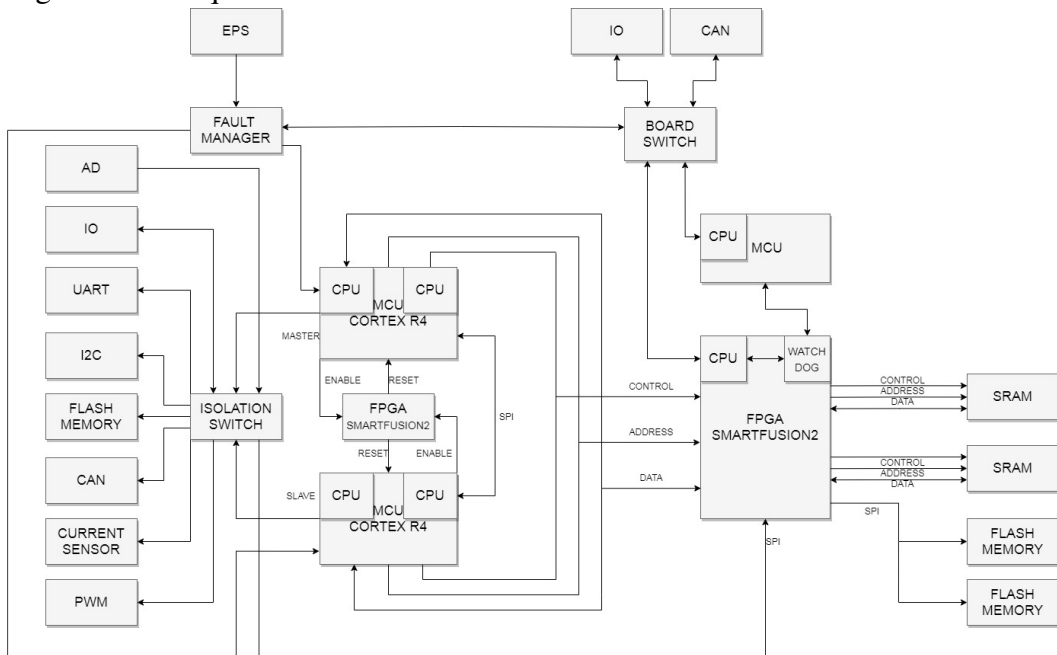
Essas técnicas de tolerância a falhas são as principais contribuições oferecidas por este trabalho. Técnicas essas que diferem das encontradas na literatura por vários aspectos, apontando melhorias onde há lacuna e englobando tudo em uma arquitetura robusta e de alta confiabilidade capaz de se proteger contra o ambiente hostil como o espacial, no qual é aceitável um pequeno período de tempo de indisponibilidade no funcionamento do nano satélite que faz uso do padrão CubeSat.

Ainda na arquitetura do Robust OBC haverá um sistema de alimentação inteligente e robusto no qual serão utilizados um conjunto de reguladores ligados diretamente nas saídas não reguladas das baterias do nano satélite. Isso permite que o subsistema do *Electrical Power System* (EPS) possa ser reiniciado sem comprometer o funcionamento do OBC, e que flutuações e ruídos gerados nos outros subsistemas não causem problemas na operação desse módulo.

Além de todos os blocos mencionados acima, o Robust OBC oferece uma gama de interface de comunicação tais como I2C, UART, CAN, *Analog to Digital Converter* (ADC), *Pulse Width Modulation* (PWM) e *Input Output* (IO) de uso geral. A figura 33 ilustra a arquitetura geral do Robust OBC.

Nas subseções seguintes, serão detalhados cada bloco dessa arquitetura apontando as contribuições científicas e as melhorias implementadas para contemplar as lacunas encontradas na literatura.

Figura 33 – Arquitetura de *hardware* do Robust OBC.



Fonte: O autor.

3.1.1 Sistema de Troca de Processador - PSS, Processor Switch System

Foi verificado nas subseções 2.4.1 e 2.4.2 que um sistema de redundância de processador é muito importante, pois ele é capaz de se precaver contra falhas transitórias e permanentes nesses dispositivos.

A análise feita em (BUSCH; SCHILLING, 2013) mostra que esse sistema de troca de processador é limitado dado que não há capacidade de ser realizado uma reconfiguração e que é dependente de componentes eletrônicos específicos para o funcionamento do chaveamento e, com isso, acabam se tornando um ponto único de falhas diminuindo a confiabilidade desse sistema.

Já em (LEPPINEN *et al.*, 2014), o sistema de troca se mostra mais eficiente e menos vulnerável, porém faz uso de um processador extra para a execução do chaveamento de processadores aumentando assim o custo e o consumo de potência desse OBC. Outra lacuna encontrada nesse sistema é a falta de proteção no processador de supervisão, pois o mesmo ainda que respaldado pela organização CubeSat ainda é susceptível aos fenômenos pertinentes ao espaço.

Após um estudo aprofundado sobre como implementar um Sistema de troca de processador que pudesse funcionar de forma precisa e ainda ser tolerante a falhas, foi verificado que o uso de uma FPGA para realizar a tarefa de supervisor seria a melhor solução visto que esse

componente é capaz de realizar reconfiguração e ainda poderá ter outras funcionalidades dentro da arquitetura dada a sua versatilidade em sistemas de computação. Como visto na seção 2.1, a FPGA Smartfusion2 do tipo Flash é adequada para ser utilizada nesse tipo de aplicação.

A função do Sistema de troca de processador é detectar falhas transitórias e permanentes nos processadores, e realizar a troca para o processador secundário, de forma que o OBC tenha chance de recuperar seu funcionamento. Esse sistema possui quatro situações de verificação de falhas, que serão expostas a seguir.

A primeira situação de falha verificada é a de travamento do processador, que será supervisionada por *watchdogs*. Esse tipo de falha pode acontecer por diversas formas, desde erros no *software*, inversão de bits em registradores importantes, ou até mesmo por falhas funcionais. Nessa situação, o processador ativo, que será chamado de *master*, ficará encarregado de enviar periodicamente dois sinais de vida para dois *watchdogs* independentes, bastando apenas um sinal de vida para manter o processador como *master*. A interrupção dos envios desses sinais de vida é considerada como uma falha do processador. Caso isso ocorra, o PSS reinicia o processador *master* e espera sua recuperação, caso a recuperação não ocorra o PSS tenta mais quatro vezes e em seguida, em caso de não recuperação, realiza a troca para o processador secundário. Os sinais dessa função são duplicados para garantir que caso ocorra falha em uma desses pinos de I/O (*stuck-at-zero*, *stuck-at-one*), o processador possa continuar em uso.

A segunda situação de falha é a de uma falha interna, detectada pelo próprio processador. A arquitetura Hercules dos processadores possui mecanismos de detecção de falhas durante execução e de *Built-In Self Test* (BIST), que permitem que o processador consiga detectar falhas decorrentes de SEUs e alguns tipos de falhas permanentes durante sua inicialização e seu funcionamento. Normalmente quando ocorrem erros durante a operação do processador, ele entra em um estado de incerteza, que dependendo de como a falha ocorra, seja possível realizar ações de recuperação ou não. Caso não seja possível retomar o funcionamento, o processador pode ativar o sinal de nERROR, que indica a FPGA que o *master* não tem condições de continuar. No caso de falhas permanentes, a ativação do sinal nERROR é dada logo no *self-test* na inicialização, evitando que um processador avariado tome posse do OBC. Como o sinal nERROR é ativado sempre na inicialização do processador, um sinal extra de nERROR MASK foi adicionado para que a FPGA entenda que a ativação do nERROR foi gerada apenas pela inicialização do processador e não uma falha.

A terceira situação é de sobrecorrente, que são causadas por SEL nos transistores do

processador. Problemas com SEL são comuns em componentes eletrônicos do tipo CMOS que não possuem proteção contra radiação, e podem ser detectadas tanto pela perda de funcionamento quanto pelo altíssimo consumo de corrente. Se essa condição não for controlada ela pode causar *burnout*, que é uma falha permanente. Para verificar esse tipo de falha, um sensor de corrente é posto nas trilhas de alimentação dos processadores, e o limite máximo de consumo é configurado no sensor através de um resistor *shunt*. Esse limite deve ser estimado por meio de testes em terra, verificando o consumo máximo de corrente dos processadores em situações semelhantes as do espaço, como por exemplo, em alta temperatura. Com isso, caso haja uma situação de sobrecorrente, um sinal de alarme é enviado para a FPGA, que desligará a alimentação dos processadores. Foi estimado um tempo de cinco segundos entre a remoção e a inserção da alimentação do processador para realizar a proteção contra SEL. Nos casos de micro SEL (falhas que geram baixo pico de corrente, mas interrompem o funcionamento do processador), eles serão detectados pelo *watchdog*.

A quarta situação é a de sobrevoltagem ou subvoltagem das fontes de alimentação do processador, que podem ocorrer caso haja algum mal funcionamento nas fontes de alimentação, gerando riscos de queima nos componentes alimentados por elas (AMEEL D. AMIDEI, 2015).

Resumidamente, essas serão as situações abordadas:

1. O *master* trava e não consegue enviar o sinal de vida para o *watchdog* (sinal WDI não atribuído no período especificado);
2. O *master* detecta uma falha, mas não é possível se recuperar (sinal nERROR é enviado pelo *master*);
3. Existe uma situação de sobrecorrente nas trilhas de alimentação dos processadores (sinal OVERCURRENT ALARM é enviado pelo sensor de corrente);
4. Existe uma situação de sobrevoltagem ou subvoltagem nas trilhas de alimentação dos processadores (sinal VOLTAGE ALARM é enviado pelo sensor de voltagem).

Dessas falhas, apenas as de voltagem não são indicadoras de falha no processador. Nesse caso a FPGA irá desligar as chaves redundantes de alimentação para proteger e evitar erros por conta das fontes de alimentação. Se essa condição persistir por muito tempo, a FPGA enviará um sinal indicando que as fontes estão defeituosas, e que a troca de placas é necessária.

Já os outros casos são indicadores de problemas no *master*, mas isso por si só não é suficiente para diferenciar se eles são causadas por falhas transitórias ou permanentes. Neste caso, só é possível determinar que o processador tem um defeito permanente caso ele continue

sem conseguir retomar sua funcionalidade mesmo após as ações de recuperação tomadas pela FPGA citadas anteriormente.

Com relação à lógica de troca dos processadores, pode-se considerar duas possibilidades. A primeira seria uma abordagem ingênua, que no caso de receber um sinal de falha do processador ativo, a FPGA apenas realizaria a troca para o secundário, sem questionar se a falha ocorrida foi transitória ou permanente. Essa abordagem é simples de implementar e garante o funcionamento do sistema enquanto tiver pelo menos um processador funcional. Se ambos os processadores estiverem com falhas permanentes torna-se necessário trocar de placa, o que pode ser detectado verificando as repetidas trocas dos processadores em um curto período de tempo.

Uma segunda possibilidade seria uma abordagem mais inteligente, de observar o processador após a detecção de uma falha, para descobrir se ele sofreu uma falha permanente ou não. Como mencionado anteriormente, essa distinção seria feita verificando se o processador continua falhando após a primeira ocorrência. Caso o *master* não consiga reestabelecer sua operação e continuar ativando os alarmes após várias tentativas, ele é considerado permanentemente danificado e é realizada a troca para o processador secundário. Essa abordagem facilita a lógica para substituição de placas, já que basta o segundo processador falhar permanentemente para iniciar a troca.

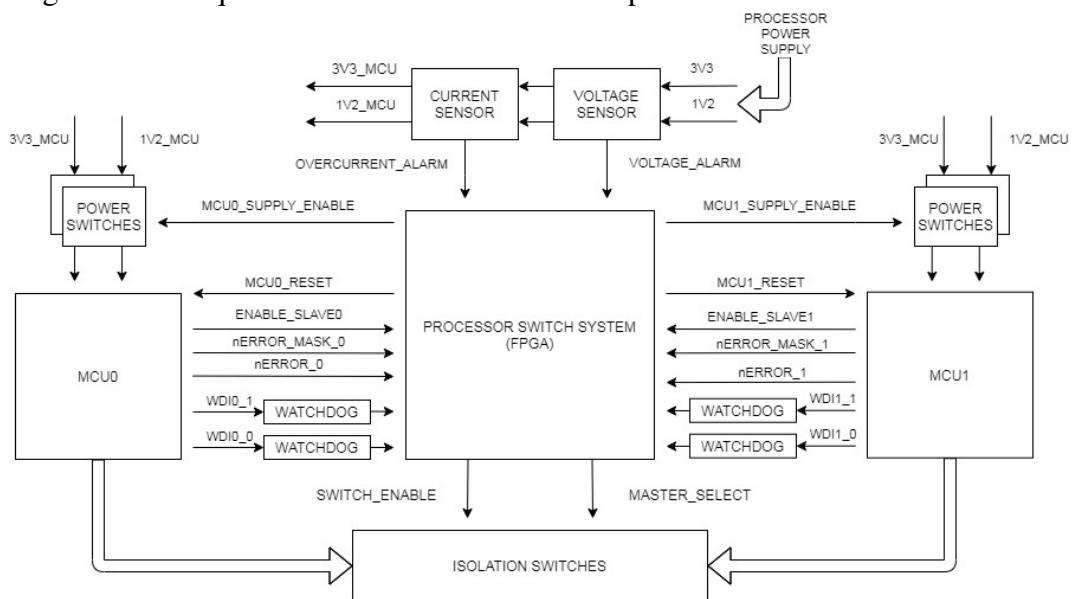
Nessa segunda abordagem, para as falhas reportadas pelo *watchdog* e pelo sinal *nERROR*, a primeira ação seria de realizar uma reinicialização quente (reiniciar o processador apenas pelo sinal de *reset*) no *master*, para permitir que o processador consiga recuperar informações da falha, o que é possível apenas com essa reinicialização. Após a primeira vez, são executados reinicializações a frio, que são feitos desativando a alimentação do processador por meio das chaves redundantes de alimentação e desativando as chaves de isolamento que são usadas para as interfaces de comunicação. Esse tipo de reinicialização é capaz de resolver mais situações de falha. No caso das falhas reportadas pelo sensor de corrente apenas reinicializações a frio são executadas, pois apenas eles são efetivos contra SEL.

Independente da lógica escolhida para o sistema de troca de processador, a troca para o processador secundário é feita da seguinte forma: a FPGA desliga as chaves de isolamento e os processadores por meio das chaves redundantes de alimentação, realiza a troca do controle do barramento para o *slave*, e depois ativa o processador, permitindo que ele retome o funcionamento do OBC. A alimentação do antigo *master* é desabilitada para que não haja consumo de corrente desnecessário. Nos dois casos, se ambos os processadores falharem, o sistema de troca de

processador enviará internamente um sinal na FPGA para indicar que o sistema está inoperante.

Uma funcionalidade adicional é que o *master* possa habilitar o funcionamento do *slave* se necessário, o que possibilita a realização de testes, atualizações de *software* ou até mesmo para processamento de dados. Por isso, os processadores terão comunicação *Serial Peripheral Interface* (SPI) entre si. Nesse caso, o *master* deve enviar o sinal ENABLE SLAVE para a FPGA, que ativará a alimentação e removerá o *slave* do modo de *reset*, sem alterar o controle do barramento. Segue, na figura 34 a ilustração da arquitetura do Sistema de troca de processador.

Figura 34 – Arquitetura do Sistema de troca de processador do Robust OBC.



Fonte: O autor.

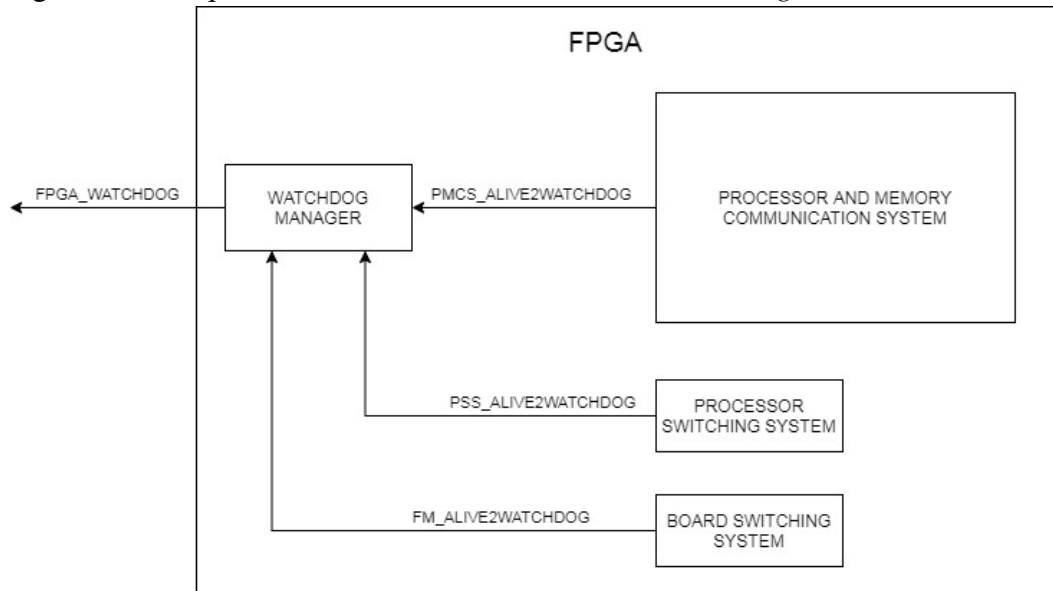
3.1.2 Gerenciador de Falha - FM, Fault Manager

Além dos processadores, é necessário observar as falhas que ocorrem no resto do sistema, principalmente as falhas na FPGA que possuem funções essenciais para funcionamento do OBC. Para realizar essa verificação será usado um sistema de gerenciamento de falhas tanto para falhas transitórias como para falhas permanentes.

A primeira situação verificada é a de interrupção de funcionamento dos módulos da FPGA, que podem acontecer de diversas maneiras, desde erros nas máquinas de estado, inversão de bits em registradores importantes, ou até em condição de SEL. Para isso, o Sistema de troca de processador, o Sistema de comunicação do processador e da memória e o Sistema de troca de OBC devem ter, em seus estados, um gerador de sinal de vida, que será processado na FPGA e

enviado para *watchdogs* externos redundantes quando ambos forem recebidos. Este módulo é chamado de Gerenciador de *watchdog* e sua representação simplificada pode ser vista na figura 35.

Figura 35 – Arquitetura do módulo Gerenciador de *watchdog* do Robust OBC.



Fonte: O autor.

A segunda situação é a de falhas detectadas durante o funcionamento da FPGA, como por exemplo, indicação do Sistema de troca de processador de que ambos os processadores estão falhando, ou das memórias possuírem falhas permanentes. Essas falhas estarão representadas pelo envio do sinal OBC FAIL ao Gerenciador de falhas.

A terceira situação é a de sobrecorrente nas trilhas de alimentação da FPGA, que podem ser geradas por eventos de SEL durante a sua operação. A abordagem será semelhante a dos processadores, utilizando um sensor de corrente e observando o seu sinal de alarme de acordo com o limite de corrente configurado. Caso isso ocorra, o sensor de corrente envia um sinal para o Gerenciador de falhas que irá remover toda a tensão de alimentação da placa e reergizá-la novamente após 10 segundos.

A quarta situação é a de sub ou sobrevoltagem nas fontes de alimentação da FPGA, que podem ocorrer devido a partículas de alta energia que causam perturbações nos circuitos analógicos e digitais das fontes, gerando riscos de queima ou mal funcionamento do sistema (AMEEL D. AMIDEI, 2015).

As falhas abordadas pelo Gerenciador de falhas serão:

1. Algum módulo interno da FPGA falha e não consegue enviar o sinal de vida para

o *watchdog*;

2. Existe algum problema grave detectado pelos módulos implementados na FPGA;
3. Existe uma situação de sobrecorrente nas trilhas de alimentação da FPGA;
4. Existe uma situação de sub/sobrevoltagem nos reguladores da FPGA.

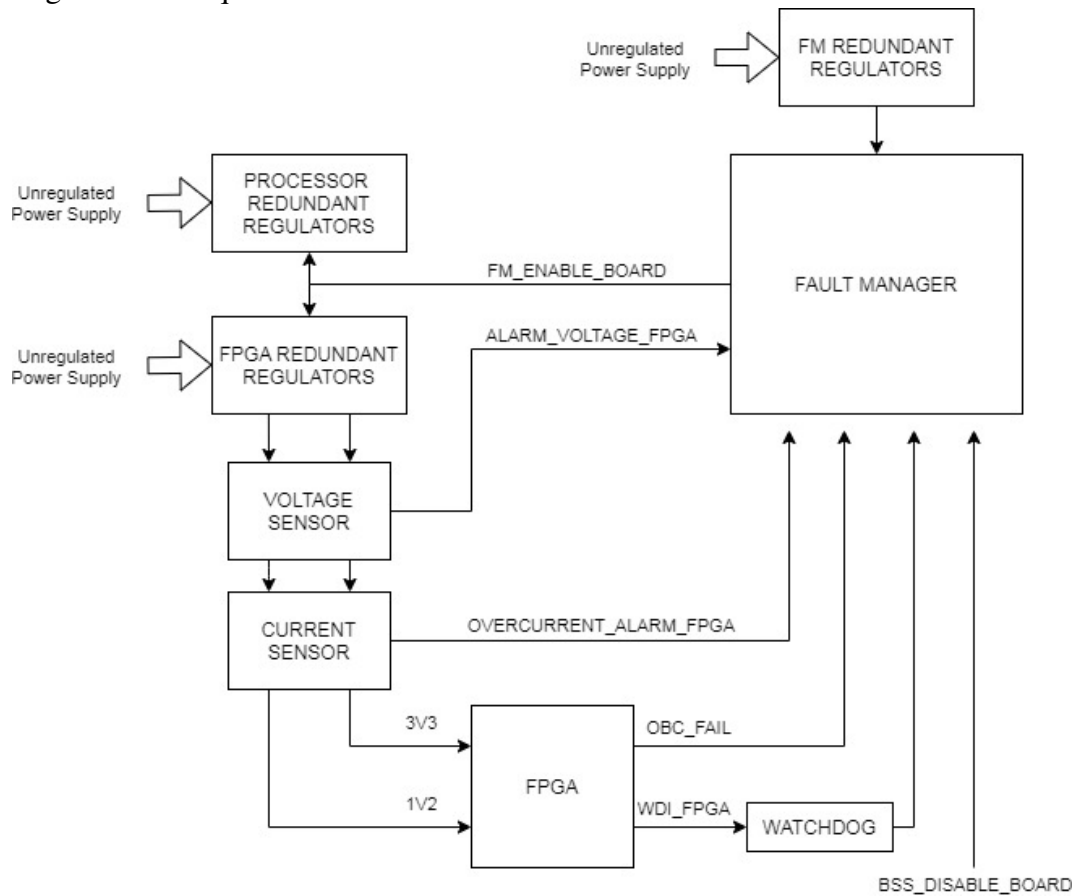
Como mencionado na descrição inicial da arquitetura do Robust OBC, o sistema será alimentado por um conjunto de reguladores ligados diretamente nas saídas não reguladas das baterias, para que seja possível reiniciar o EPS sem comprometer o funcionamento do OBC, e que flutuações e ruídos gerados nos outros subsistemas não causem problemas na operação do módulo de computação.

Em todos esses eventos, a abordagem de funcionamento escolhida é de desativar diretamente os reguladores da placa, retirando completamente a alimentação dos circuitos do OBC por um período de tempo controlado por um temporizador. Esta é a última alternativa disponível para proteger o sistema de falhas de alimentação e reiniciar os módulos para um estado inicial seguro. Assim como no Sistema de troca do processador, essas quatro situações de falha podem ser causadas tanto por falhas transitórias ou permanentes, e a diferenciação entre elas é dada pela quantidade de falhas ocorridas repetitivamente. Caso esse recurso seja utilizado várias vezes consecutivas, a placa é considerada defeituosa, e a partir disso a alimentação do sistema é desabilitada e o controle do nano satélite é dado à outro OBC, pelo Sistema de troca de OBC. A arquitetura geral do Gerenciador de falhas é ilustrada na figura 36.

3.1.3 Sistema de Troca de Placa - BSS, Board Switch System

O Sistema de troca de placa foi concebido para possuir até três OBCs e tem como idéia principal a utilização de um barramento dedicado entre os computadores de bordo do nano satélite, com sinais de controle e de comunicação para detectar a interrupção de funcionamento do OBC ativo. O sistema é baseado na detecção de falhas de operação a partir do sinal de controle de fonte (FM ENABLE BOARD), no qual os computadores de bordo sobressalentes recebem essa informação, e realizam uma comunicação entre si utilizando a interface CAN. Para realizar a troca desta forma, seria necessário um sistema processado para controlar essa comunicação. De acordo com (VINCI; SAOTOME, 2010), a redundância de *hardware* do sistema de supervisão de troca de OBC possui uma maior confiabilidade em comparação com o uso de somente um componente atuando como supervisor. Diferentemente de trabalhos como (FAYYAZ *et al.*, 2012) que possui um sistema de troca de OBC implementado com somente um componente eletrônico,

Figura 36 – Arquitetura do Gerenciador de falhas do Robust OBC.



Fonte: O autor.

acarretando em um SPF, o Robust OBC possui uma redundância no Sistema de troca de placa. Neste caso, teria a possibilidade de utilizar o processador embutido na FPGA e um processador externo.

Com isso, o Sistema de troca de OBC é baseado no uso de dois processadores, um externo que fica responsável por gerenciar a comunicação enquanto as fontes de alimentação do OBC permanecerem desativadas, e o interno da FPGA. Ambos os processadores tem a capacidade de indicar a necessidade de desativação do sistema para o Gerenciador de falhas por meio dos sinais FM ENABLE BOARD, que controlam as fontes de alimentação. É implementado na FPGA um módulo de controle dos processadores denominado Sistema de troca de placa, que dentre as suas diversas funções, possui a capacidade de desativar o processador externo por meio do sinal BSS RST MCU1, para diminuir o consumo do sistema.

Assim como os sistemas das subseções 3.1.1 e 3.1.2, o Sistema de troca de placas também possui quatro situações de falhas nas quais esse sistema atua, porém a troca somente se dará em falhas do tipo permanente nos componentes críticos do OBC.

A primeira situação é decorrente da falha permanente do próprio Sistema de troca

de placa. Nesse caso os dois processadores redundantes do sistema deixam de funcionar permanentemente e faz com que o segundo OBC assuma o controle através do barramento de comunicação I/O BSS. Esta situação de falha é a única que inviabiliza a troca de contexto atual do OBC, dado que o barramento CAN estará indisponível.

A segunda situação acontece quando a FPGA falha permanentemente ou por TID, ou por SEL, ou por causa do aumento da tensão no sistema de alimentação da FPGA, o que acarreta na queima da FPGA. Dessa forma o Gerenciador de falhas receberá um sinal ALARM VOLTAGE FPGA, ou OVERCURRENT ALARM FPGA, ou WATCHDOG FPGA, e assim o processo de troca de OBC é realizado.

A terceira situação é dada através da falha permanente dos processadores do OBC. Essa situação se dá por problemas como TID e SEL que acabou não sendo protegida pelo Sistema de troca de processador, ou pelo mal funcionamento do sistema de alimentação. Com isso, o Sistema de troca de placa é ativado através do sinal OBC FAIL.

A quarta e última situação advém da falha permanente das memórias SRAM. Assim como as outras situações citadas, a forma dessa falha ocorrer acontece ou por TID, ou por SEL, ou falha no sistema de alimentação e assim como a terceira situação, o sinal OBC FAIL é ativado e a troca de placa é realizada.

Em resumo, as situações de ocorrência de troca de OBC são:

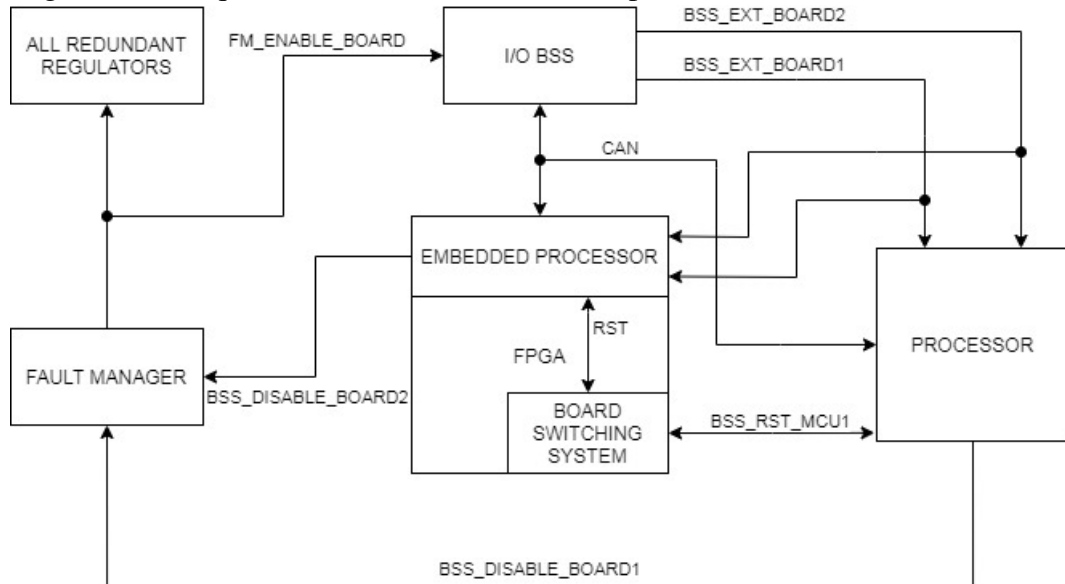
1. O Sistema de troca de placa falha permanentemente;
2. A FPGA falha permanentemente;
3. Os processadores redundantes a frio falham permanentemente;
4. As memórias SRAM falham permanentemente.

O funcionamento do Sistema de troca de placa acontece da seguinte maneira: Ao detectar necessidade de troca de placas, o Sistema envia um comando que ligará a fonte do OBC subsequente e em seguida se comunicará com o Sistema deste OBC, através da interface CAN, para passar as informações necessárias para a inicialização deste novo OBC como, por exemplo, chaveamento de contexto. Após isso, o OBC atual desliga o anterior e então assume o comando do nano satélite.

Como mencionado anteriormente, o Sistema de troca de placa contém sinais de estado de até três computadores de bordo, o que deve ser configurado em cada placa via resistores de OR durante a montagem dos componentes eletrônicos na placa. Cada placa controla apenas um dos sinais, enquanto os restantes serão lidos pelos processadores presentes na placa (representados

na figura 37 pelos sinais BSS EXT BOARD1 e BSS EXT BOARD2), pois indicam o estado dos outros computadores de bordo. A figura 37 apresenta a arquitetura do Sistema de troca de placa.

Figura 37 – Arquitetura do Sistema de troca de placa do Robust OBC.



Fonte: O autor.

A arquitetura do Sistema de troca de placa foi concebida pensando no chaveamento de contexto entre os OBCs do nano satélite. Logo a interface de comunicação CAN se fez necessária para que essa técnica pudesse ser utilizada dando mais robustez a essa solução tolerante a falhas.

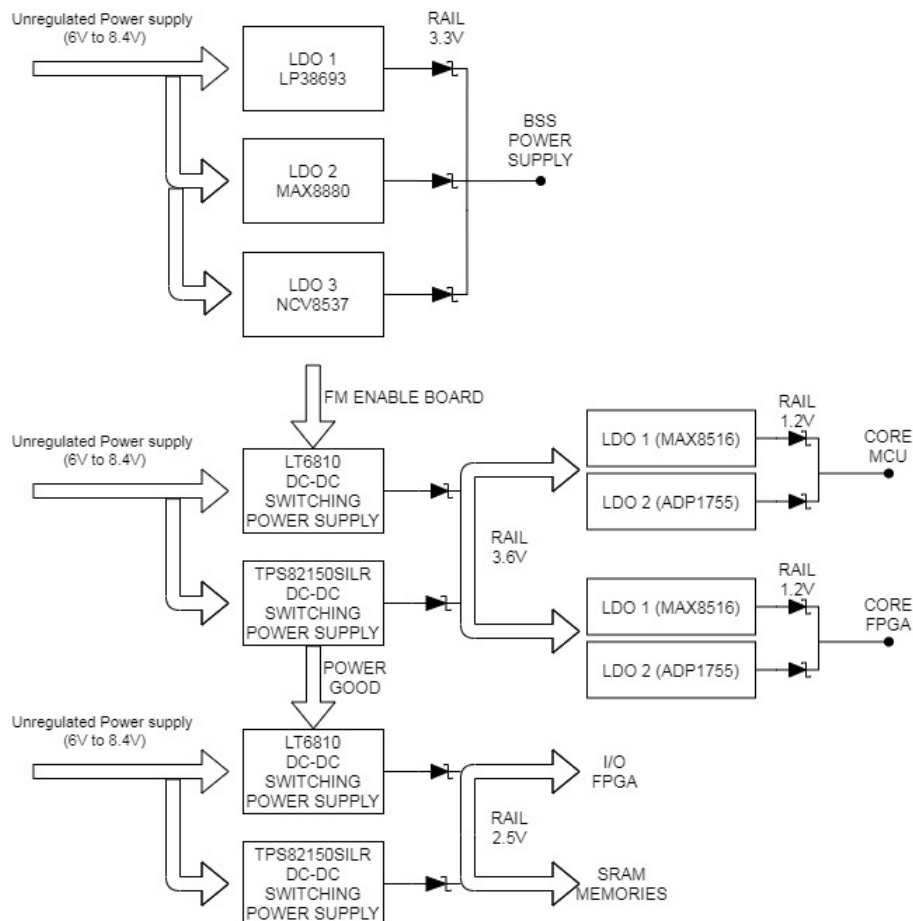
3.1.4 Sistema de Alimentação Inteligente - IPS, Intelligent Power System

O Sistema de alimentação do Robust OBC é dito inteligente pelo fato do mesmo possuir uma sequência de alimentação, ser capaz de se auto monitorar e somente liberar a próxima tensão de alimentação (através dos sinais de PGOOD) caso a tensão anterior esteja funcional e estável. Isso evita que haja, em caso de falhas das fontes redundantes antecessoras, uma cadeia de falhas provenientes da falha antecessora. Além disso, o processador externo do Sistema de troca de placas recebe o sinal PGOOD e com isso é capaz de executar a troca de placas caso perceba que as fontes estão em estado de falha. Essa característica de tolerância a falhas não foi mencionada na subseção 3.1.3 dada a sua conectividade com o Sistema de alimentação inteligente.

O funcionamento desse sistema acontece da seguinte forma: as fontes de alimentação de tripla redundância do Sistema de troca de placa e do Gerenciador de falhas são energizadas

assim que o Robust OBC é conectado na bateria do nano satélite. Em seguida, abre uma contagem de tempo de 10 segundos através de um temporizador (esse tempo foi utilizado nos testes de desenvolvimento, mas pode ser alterado conforme a aplicação) que é o tempo de reinicialização do Robust OBC para uma situação de SEL na FPGA, ou reconfiguração, ou reinicialização do sistema como um todo. Após a contagem desse tempo, as fontes redundantes de 3,3V são liberadas e, consequentemente, libera o sinal de PGOOD que, por sua vez, libera as fontes de alimentação redundantes de 2,5V. Quando as fontes de alimentação de 3,3V estão em pleno funcionamento, são ativadas em seguida, as fontes de alimentação redundantes de 1,2V. O diagrama de funcionamento do Sistema de alimentação inteligente é ilustrado na figura 38.

Figura 38 – Arquitetura do Sistema de alimentação inteligente do Robust OBC.



Fonte: O autor.

O uso da técnica de redundância ativa em todas as fontes de alimentação se deu pelo fato de que esse sistema é o mais importante do Robust OBC, simplesmente pelo fato de que se ele não funcionar, nenhum outro circuito do OBC irá funcionar e com isso, o aumento da confiabilidade se fez necessário.

Tendo em vista a importância desse sistema para o OBC, a fonte de alimentação que energiza o Sistema de troca de placa e o Gerenciador de falhas deverá possuir uma alta confiabilidade dado que esses sistemas são os que detectam falhas e realizam o chaveamento de OBC e, assim, não inviabilize a missão do nano satélite. Dito isso, foi aplicada a técnica de redundância tripla nessa fonte de alimentação.

Além da técnica de redundância para aumentar a confiabilidade, um outro elemento é de grande importância para o acréscimo desse parâmetro que é a tolerância a TID e SEE dos reguladores de tensão que implementam as fontes de alimentação. De acordo com (AMEEL D. AMIDEI, 2015), alguns reguladores de tensão são mais susceptíveis a TID, outros a SEE. Uma boa estratégia a ser utilizada é a implementação de redundância com reguladores com susceptibilidades diferentes entre eles para que os fenômenos provenientes do espaço tenham efeitos distintos nos reguladores e assim a probabilidade de falha tenha uma menor incidência.

3.1.5 Sistema de Comunicação do Processador e da Memória - PMCS, Processor and Memory Communication System

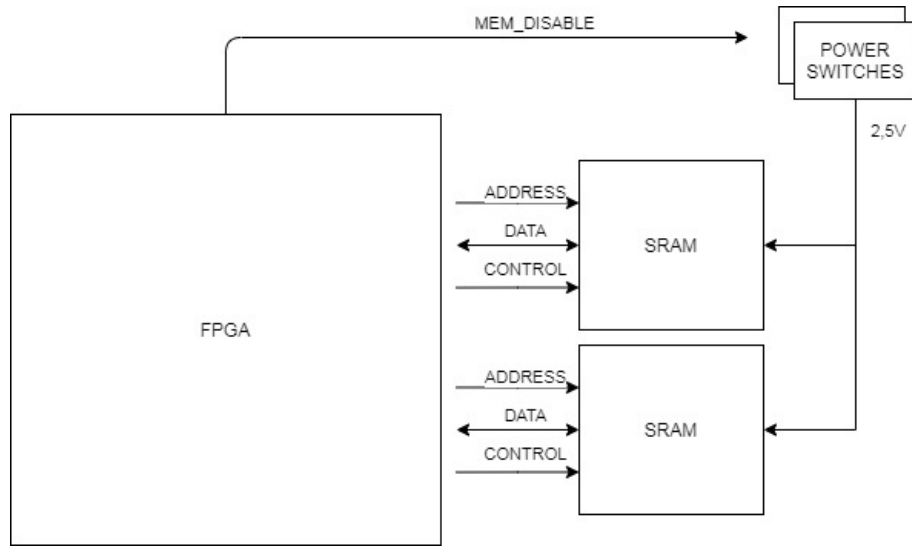
Nesse sistema, a FPGA será responsável por mediar a comunicação entre as memórias e os processadores, detectando e corrigindo erros por meio de códigos corretores de erros. Além disso, ela também será responsável por detectar falhas funcionais e de SEL nas memórias, o que será feito de duas formas:

1. A FPGA tenta realizar operações de leitura e escrita na memória, mas não obtém resposta;
2. Existe uma situação de sobrecorrente nas trilhas de alimentação das memórias.

Em ambos os casos, a abordagem para solucionar esses problemas é de reiniciar as memórias, interrompendo a sua alimentação por meio de chaves redundantes, e colocando os sinais de controle entre a FPGA e as memórias em nível lógico baixo. Isso ocasionará perda de dados, mas é o único meio disponível para corrigir esses estados de falha.

Assim como nos processadores, essas duas situações de falha podem ser causadas tanto por falhas permanentes quanto por falhas transitórias. Só é possível determinar que as memórias possuam um defeito permanente caso elas continuem sem conseguir retomar sua funcionalidade mesmo após as ações de recuperação. Neste caso, será gerado um sinal indicando que as memórias falharam permanentemente, e que a troca de placas é necessária. A figura 39 ilustra o a arquitetura para falhas permanentes e funcionais das memórias SRAM.

Figura 39 – Arquitetura para falhas permanentes e funcionais das memórias SRAM.



Fonte: O autor.

Como dito anteriormente, a remoção de alimentação faz com que haja perda de dados pela característica volátil da memória SRAM, logo essa técnica é ineficiente contra inversões de bits nos dados armazenados nas memórias SRAM provenientes de um SEU. Com isso, se faz necessário a utilização de códigos corretores de erros para enrobustecer todo o Sistema de comunicação do processador e da memória e, assim contemplar todas os principais problemas pertinentes aos efeitos naturais do ambiente espacial.

O principal problema quando se é utilizado EDAC para a proteção de memórias é a redundância associada. Os bits extras adicionados usados para detectar e/ou corrigir os possíveis erros ocorridos aumentam o espaço necessário na utilização da memória. Dessa forma, a quantidade de armazenamento ocupada para bits redundantes é dimensionada com a capacidade de memória. Por exemplo, se um EDAC com 100 por cento de redundância for empregado em uma memória de 2 GB, apenas 1 GB estará disponível para armazenar o código de fato, o restante 1 GB é necessário para bits de código (GRACIA-MORÁN *et al.*, 2018), com isso o custo da memória será maior.

Além desse custo, o uso de um EDAC implica em aumento de área, potência e atraso empregados pelos circuitos do codificador e do decodificador. Esses aumentos devem ser mantidas o mais baixo possível, especialmente em aplicações espaciais que possui restrições como área e potência (GRACIA-MORÁN *et al.*, 2018).

Com isso, os EDACs selecionados para serem utilizados no Robust OBC foram os do artigo (GRACIA-MORÁN *et al.*, 2018), pois eles são capazes de reduzir bastante a redundância

adicional, mantendo, ou mesmo melhorando, a cobertura de erros de memória de EDACs como (CASTRO *et al.*, 2016). Além disso, os custos gerais de área, potência e atraso também são reduzidas, satisfazendo a aplicação que serão submetidos.

Um outro fator de seleção desses códigos é que eles são complementares, assim, o uso de um dos três EDACs será feito dinamicamente de acordo com tipo de erro que será detectado.

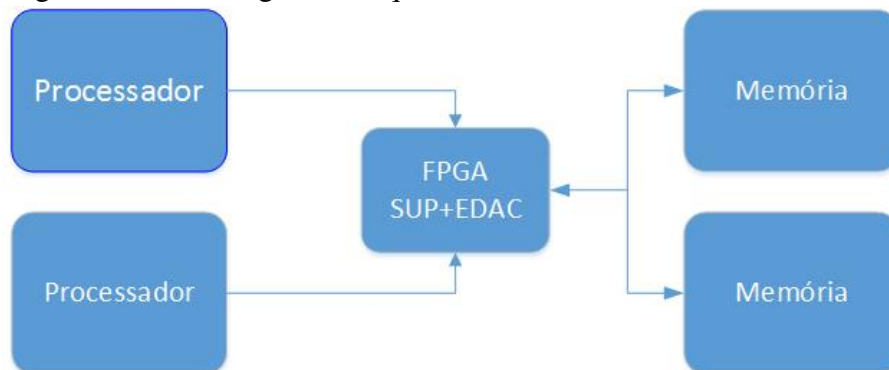
3.2 Confiabilidade

Assim como foi feito na seção 2.6, a arquitetura do Robust OBC também foi modelada para que fosse capaz de extrair as expressões matemáticas pertinentes a possíveis falhas permanentes e transitórias.

Os principais blocos que serão incluídos no modelo de arquitetura são o Sistema de troca de processador, o Sistema de troca de placa e o Sistema de comunicação do processador e da memória.

Como a FPGA tem um papel importante e distinto para todos os sistemas acima citados, ela será desmembrada para a obtenção da expressão matemática da confiabilidade do sistema. A figura 40 ilustra o modelo do Robust OBC baseado na sua arquitetura de *hardware*.

Figura 40 – Modelagem da arquitetura do Robust OBC.



Fonte: O autor.

– Confiabilidade para falhas permanentes:

Assim como o OBC 2, seu modelo de arquitetura segue o padrão 2 e a modelagem do sistema de computação do Robust OBC possui a expressão matemática a seguir:

$$R_{Robc}(t) = R_{sup} * (2e^{-\lambda_{prot}} - e^{-2\lambda_{prot}}) (1 - (1 - e^{-\lambda_{mem}})(1 - e^{-\lambda_{mem}})). \quad (3.1)$$

Onde:

R_{Robc} é a confiabilidade do Robust OBC;

R_{sup} é a confiabilidade do supervisor (FPGA);

λ_{pro} é a taxa de falha do processador;

λ_{mem} é a taxa de falha da memória.

– Confiabilidade para falhas transitórias:

Nesse modelo, a FPGA possui um duplo papel. Ela atua como supervisor no Sistema de troca de processador e atua como EDAC no Sistema de comunicação do processador e da memória. Com isso, o bloco da FPGA será duplicado para a obtenção da expressão matemática da confiabilidade para falhas transitórias. Logo, a região que contém os processadores e o supervisor será semelhante ao modelo de Markov da figura 24 e chamada de Região 1, já a região que contém o EDAC é o mesmo modelo de Markov mostrado na figura 27 e chamada de Região 2. Então a expressão matemática para falhas transitórias é dada a seguir:

$$R_{Robc}(t) = \left[\left(2 + \frac{\mu_1}{\lambda_1} \right) e^{-\lambda_1 t} - \left(1 + \frac{\mu_1}{\lambda_1} \right) e^{-2\lambda_1 t} \right] \left[\frac{1}{(\lambda_2 + \mu_2)} (\mu_2 + \lambda_2 e^{-(\lambda_2 + \mu_2)t}) \right] (1 - (1 - e^{-\lambda_{mem} t}) (1 - e^{-\lambda_{mem} t})). \quad (3.2)$$

Como a proposta do Robust OBC é de uma arquitetura modular com até três unidades de computadores de bordo, se faz necessário o cálculo da confiabilidade para falhas permanentes do sistema completo.

Conforme dito anteriormente, o Sistema de troca de placa, por ser de grande importância na solução proposta, possui redundância ativa com componentes distintos tendo objetivo de aumentar a confiabilidade desse sistema, ou seja, a representação em blocos desse módulo é replicada e mostrada na figura 41.

Os blocos em paralelo dos supervisores serão utilizados para as duas situações de falhas permanentes e falhas transitórias, portando será primeiramente obtida a expressão matemática desse bloco para depois determinar as demais equações.

Fazendo uso da Equação 2.7, a expressão matemática de confiabilidade dos blocos dos supervisores é:

$$R_{sup}(t) = (1 - (1 - e^{-\lambda_{pro1} t})(1 - e^{-\lambda_{pro2} t})). \quad (3.3)$$

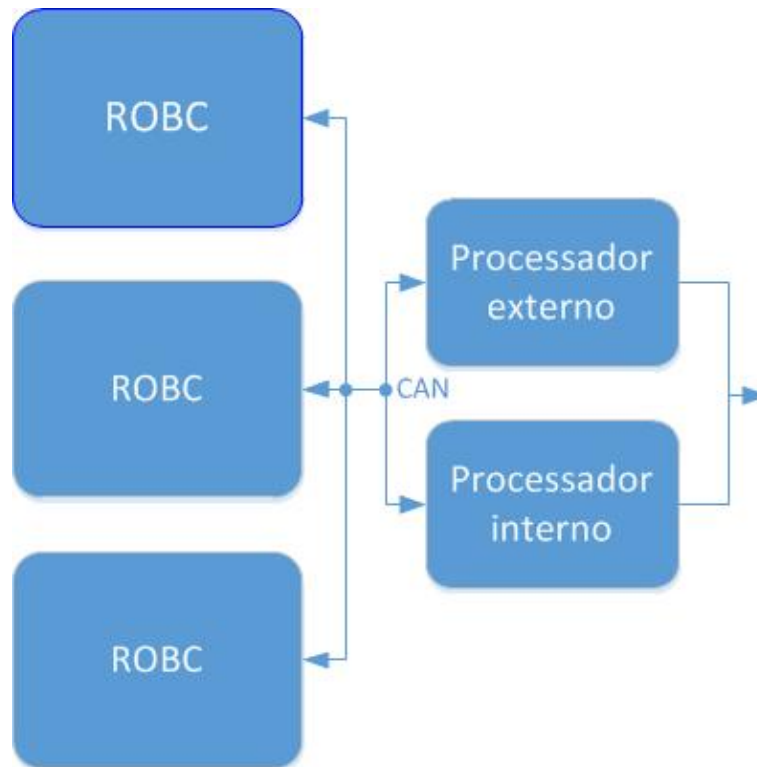
Onde:

R_{sup} é a confiabilidade da redundância do supervisor;

λ_{pro1} é a taxa de falha do processador externo;

λ_{pro2} é a taxa de falha do processador interno.

Figura 41 – Modelagem do Sistema de troca de placa do Robust OBC.



Fonte: O autor.

– Confiabilidade para falhas permanentes:

Para a situação de falhas permanentes, foi usado o modelo de TMR descrito na subseção 2.5.4 e na Equação 2.20, logo a expressão matemática é:

$$R_{Robc_{tmr}}(t) = R_{sup} * (3e^{-2\lambda_{robct}} - 2e^{-3\lambda_{robct}}). \quad (3.4)$$

Onde:

$R_{Robc_{tmr}}$ é a confiabilidade do sistema com três Robust OBC;

R_{sup} é a confiabilidade do supervisor;

λ_{robct} é a taxa de falha do Robust OBC.

3.3 Conclusão

Neste capítulo foi realizada toda pesquisa e desenvolvimento para a concepção de uma arquitetura robusta e tolerante a falhas de um computador de bordo COTS para nano satélite de padrão CubeSat.

Foram verificadas as lacunas existentes na comunidade acadêmica e no meio corporativo para propor contribuições relevantes e inovadoras que pudessem agregar valor a organização

CubeSat que é uma comunidade relativamente nova com várias oportunidades de pesquisa científica, porém pequena e restrita.

Com isso, foram demonstradas todas as idéias concebidas a nível de arquitetura ilustrando as técnicas de tolerância a falhas empregadas e apontando as melhorias e benefícios do uso dessas técnicas no computador de bordo Robust OBC.

Por fim, foram obtidas as expressões matemáticas da confiabilidade para falhas permanentes e transitórias do sistema de computação da arquitetura proposta com objetivo de realizar uma comparação entre o Robust OBC e os outros OBCs discutidos nessa tese, levando em consideração o valor obtido de confiabilidade e as técnicas empregadas em cada OBC.

No próximo capítulo, será aplicada a técnica de FMEA no contexto de diagrama de falhas como forma de resultado de bom funcionamento das lógicas propostas nos sistemas que compõem a arquitetura do Robust OBC. Em seguida, será mostrado o resultado físico da arquitetura concebida em forma de placa de circuito impresso. Outros resultados relevante são os testes realizados e aprovados de todos os sistemas discutidos, analisando seu desempenho em conjunto com o consumo de energia dissipado.

4 RESULTADOS

Após a concepção da arquitetura do Robust OBC, uma série de atividades fizeram-se necessárias para validação das idéias de contribuição estabelecidas nesta tese.

A primeira é a análise de falhas da arquitetura. Através de um diagrama de falhas, será visto como o sistema deverá se comportar dependendo de como seja a falha. Toda essa análise será feita de forma teórica.

A segunda é a materialização da arquitetura em forma de placa de circuito impresso. Para a validação na prática do funcionamento de todas as técnicas de tolerância a falhas empregadas no Robust OBC, uma PCI se fez necessária para suportar todos os circuitos eletrônicos que implementam as técnicas citadas.

A terceira é o teste para verificação do funcionamento dos sistemas concebidos na fase de arquitetura. Serão validados todos os sistemas e nessa tarefa serão demonstradas a implementação realizada, o *setup* de teste no qual o Robust OBC foi submetido e a análise dos pontos positivos e negativos juntamente com a associação do funcionamento do sistema com o consumo de corrente dissipada.

A quarta e última é a comparação da confiabilidade e das técnicas utilizadas em todos os OBCs discutidos nessa tese com o Robust OBC, através de gráficos e tabelas .

4.1 Análise de falhas

A classificação de falhas e erros com base em sua duração serve para um propósito útil. A abordagem tolerar uma falha depende da sua duração. Tolerar uma falha permanente requer a capacidade de evitar o uso do componente defeituoso, talvez usando uma réplica desse componente sem falhas. Tolerar uma falha transitória não requer esse reparo próprio, porque a falha não persistirá. Esquemas de tolerância a falhas tendem a tratar falhas intermitentes como transitórias ou permanentes, dependendo da frequência com que ocorrem, embora existam alguns esquemas projetados especificamente para tolerar falhas intermitentes (DUBROVA, 2013).

- Transitória: Uma falha transitória ocorre uma vez e depois não persiste. Um erro devido a uma falha transitória é geralmente chamado de *soft error* ou SEU.
- Permanente: Uma falha permanente ocorre em algum momento, talvez até introduzido durante a fabricação de chips e persista a partir desse momento ou mesmo resultado de um SEL que destruiu o componente/sistema. É provável que uma única falha permanente se

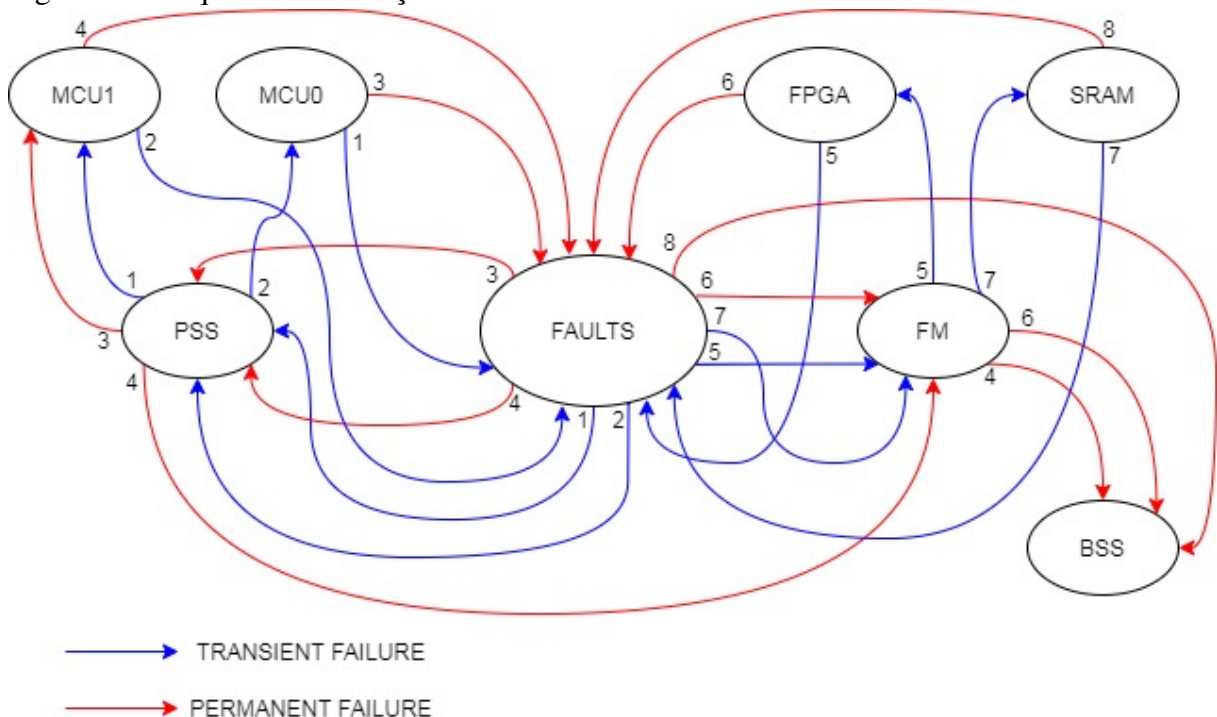
manifeste como um erro repetido, a menos que o componente defeituoso seja reparado, porque o componente defeituoso continuará sendo usado e produzirá resultados incorretos.

- Intermitente: Uma falha intermitente ocorre repetidamente, mas não continuamente, da mesma maneira. Como tal, uma falha intermitente se manifesta através de erros intermitentes.

No caso deste trabalho, os casos para falhas intermitentes serão tratados como casos de falhas permanentes de acordo com o conceito acima citado.

A análise de falhas, que é uma técnica do FMEA, do projeto Robust OBC é feita verificando as possíveis falhas que poderão ocorrer nos componentes críticos do sistema. Técnicas de tolerância a falhas como redundância, sistema de troca de processador, sistema reconfigurável e códigos corretores de erros foram utilizados como forma de mitigar e suportar falhas permanentes e transitórias. A figura 42 ilustra o esquema de transição de falhas entre os componentes críticos existentes no RobustOBC.

Figura 42 – Esquema de transição de falhas do Robust OBC.



Fonte: O autor.

O fluxo das setas na figura 42 é representado por numeração na forma crescente. Este fluxo se dá da seguinte maneira, a primeira análise é feita do ponto de vista da cada componente crítico (a análise dos processadores é feita em conjunto), onde, primeiramente, é realizado o caminho da falha transitória e por fim a falha permanente. Tomando como exemplo o início

dos fluxos (caminho 1), o MCU0 tem uma falha transitória e logo aciona o Sistema de troca de processador que, por sua vez, faz a troca para o MCU1. Em seguida (caminho 2), o MCU1 tem uma falha transitória, que passa pelo Sistema de troca de processador, no qual o mesmo aciona o MCU0. Após a análise de falha transitória, é feita a análise de falha permanente que é iniciada pelo caminho 3. O MCU0 obtém uma falha permanente e o Sistema de troca de processador é acionado, que por sua vez, aciona o MCU1. Quando o MCU1 tem uma falha permanente, o Sistema de troca de processador não mais poderá realizar a troca de processadores pelo fato do MCU0 já estar em falha permanente, logo seu fluxo (caminho 4) será o da troca de placa pelo Sistema de troca de placa. E assim são feitos todos os fluxos existentes na figura 42.

Como dito logo acima, toda a análise de falhas é feita em cima dos componentes críticos que são os processadores, a FPGA e as memórias SRAM. Com isso, serão descritos, para cada componente e tipo de falha, a possível causa da falha, o que acarreta no componente e a ação que se deve tomar para obter tolerância a falhas no sistema.

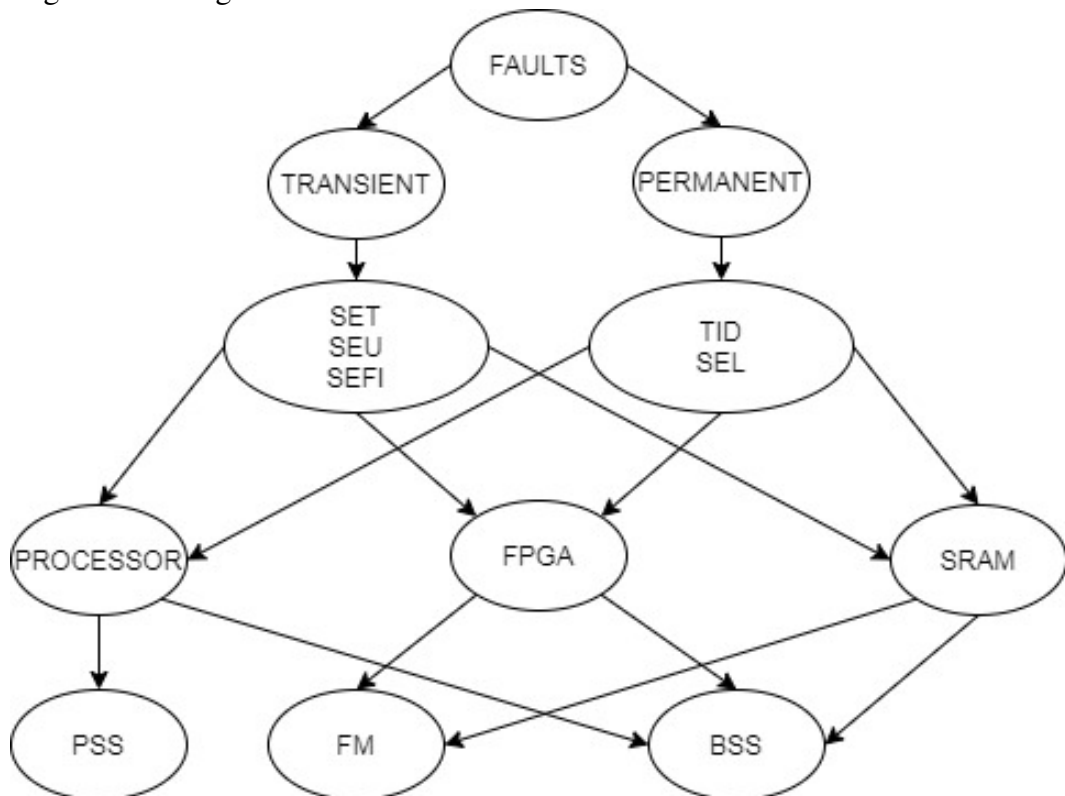
- Processador:
 - Falha transitória: Pode ser proveniente de SET, SEU ou SEFI que acarreta em inversão de bits e em falhas funcionais;
 - Ação: Troca de processador para que o processador anterior se recupere. Poderá se recuperar através de reinicialização ou desligando sua alimentação e ligando novamente;
 - Falha permanente: Pode ser proveniente de TID ou SEL que acarreta na queima parcial ou total do componente;
 - Ação: Troca do processador, caso o segundo também esteja em estado de falha permanente haverá, então, a troca de placa.
- FPGA:
 - Falha transitória: Pode ser proveniente de SET, SEU ou SEFI que acarreta em inversão e em falhas funcionais;
 - Ação: Enviar um sinal para o Gerenciador de falhas que irá desligar a alimentação. Poderá se recuperar ligando a alimentação novamente;
 - Falha permanente: Pode ser proveniente de TID ou SEL que acarreta na queima parcial ou total do componente;
 - Ação: Haverá a troca de placa.
- Memória SRAM:

- Falha transitória: Pode ser proveniente de SET, SEU ou SEFI que acarreta em inversão e em falhas funcionais;
- Ação: Haverá códigos corretores de erros adequados para alguns tipos de erro que possivelmente ocorrerão;
- Falha permanente: Pode ser proveniente de TID ou SEL que acarreta na queima parcial ou total do componente;
- Ação: Haverá a troca de placa.

Caso haja SEL em algum dos componentes críticos, o sensor de corrente é acionado e fará o corte de alimentação do componente em questão. Em seguida ele será energizado novamente para analisar se o mesmo entrou ou não em situação de falha permanente.

A figura 43 ilustra a descrição de falhas e tomada de decisão para os mais variados efeitos que possam vir a surgir em cada componente crítico do Robust OBC.

Figura 43 – Diagrama de falhas do Robust OBC.



Fonte: O autor.

4.2 Placa de Circuito Impresso do Robust OBC

Um resultado importante desse trabalho foi a materialização da arquitetura proposta em forma de placa de circuito impresso. Porém para alcançar esse objetivo, foi necessário realizar algumas outras atividades como selecionar os componentes críticos e desenhar os esquemas elétricos que implementam a arquitetura do Robust OBC.

Como dito na seção 2.1, a FPGA escolhida para o Robust OBC foi a M2S010-VFG400I de 65 nm do fabricante Microsemi (MICROSEMI, 2018), dada sua implementação do tipo Flash e suas tolerâncias a SEU, SEL e possuir alguma resistência a TID conforme (DSILVA *et al.*, 2015) e (REZZAK *et al.*, 2014). Além disso, essa FPGA já possui histórico de vôos em nano satélite, dando mais credibilidade ao componente.

Já o processador escolhido foi o TMS5701114CZWTQQ1 de 65 nm do fabricante Texas Instruments que possui diversas características de tolerância a falhas em sua construção (TEXAS INSTRUMENTS, 2015) e já é bastante usado no segmento de nano satélites (MOTA, 2017) (LEPPINEN *et al.*, 2014) (YUEN; SIMA, 2018).

A escolha da memória SRAM foi baseada na relação custo/benefício, pois será empregada na mesma diversas técnicas de tolerância a falhas como redundância, proteção contra SEL e EDACs, portanto o custo foi o fator decisivo para seu uso. Além disso, outro fator de seleção da memória foi o fato dela já ter sido testada por diversos trabalhos como (CORNETTI *et al.*, 2021) e (CECCHETTO *et al.*, 2020). Logo a memória selecionada foi a IS61WV204816BLL-10TLI de 40 nm do fabricante ISSI (ISSI, 2016).

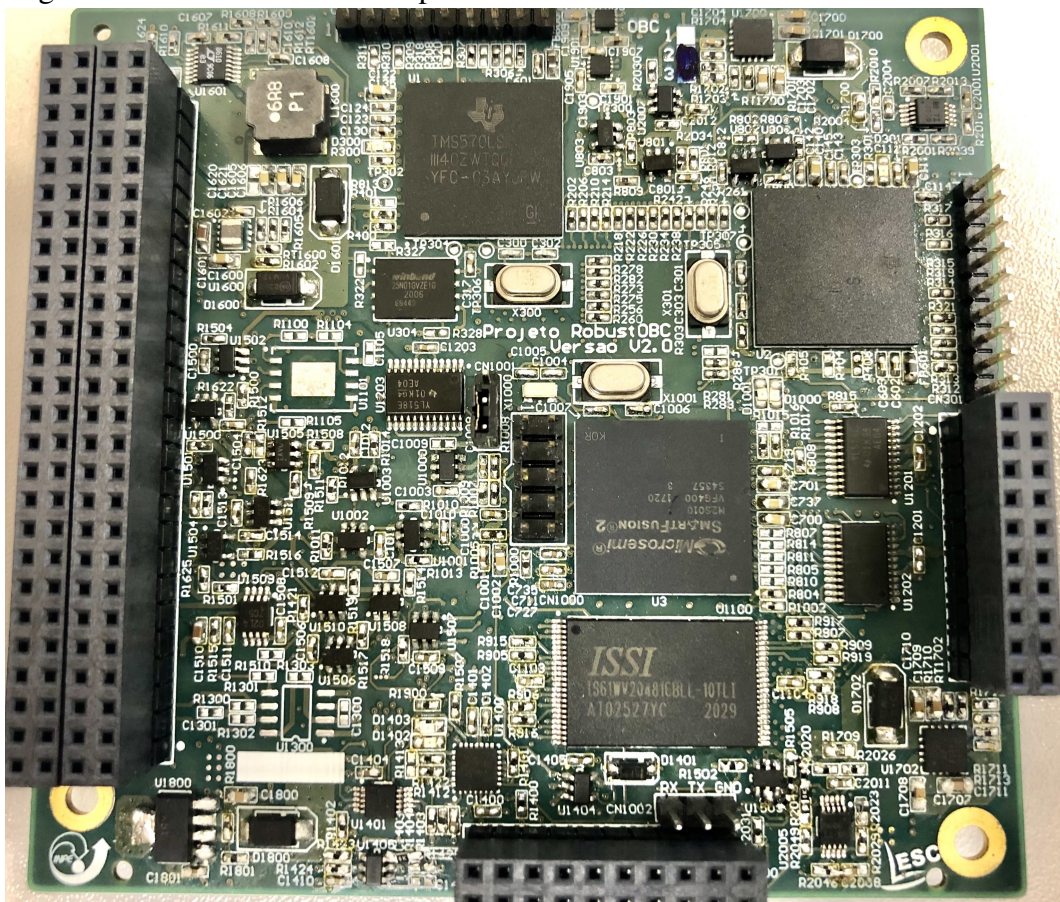
Vale ainda ressaltar as escolhas feitas nos reguladores de tensão, dada a sua importância para qualquer sistema eletrônico. A escolha se deu através de testes realizados em (AMEEL D. AMIDEI, 2015) e, com isso, foram selecionados os reguladores LT8610EMSE do fabricante Linear Technology (LINEAR TECHNOLOGY, 2021), ADP1755ACPZ-R7 do fabricante Analog Devices (ANALOG DEVICES, 2014) e o MAX8516EUB+T do fabricante Maxim Integrated (MAXIM INTEGRATED, 2016).

Após a seleção dos componentes eletrônicos, foram desenvolvidos os esquemas elétricos que implementam todas as técnicas de tolerância a falhas propostas na arquitetura. Esses esquemas foram divididos em vinte páginas baseados nas funções que os circuitos irão desempenhar. Todas as páginas dos esquemas elétricos encontram-se disponíveis no apêndice A.

Com a finalização das atividades anteriores, foi realizado o roteamento da placa de circuito impresso do Robust OBC utilizando o *software* Altium Designer. Esse roteamento foi

feito utilizando diversas técnicas de prevenção a interferência eletromagnética como roteamento de pares diferenciais nos sinais de comunicação CAN e serpentinas para equalização das trilhas dos barramentos de dados e endereço de alta velocidade na comunicação entre processador, FPGA e memória. Com isso, se fez necessário o uso de uma placa com dez camadas para comportar a grande quantidade de sinais existente no Robust OBC e ainda realizar blindagem de sinais importantes e de alta velocidade através de camadas de potencial Terra e Tensão de Alimentação. A figura 44 ilustra a PCI desenvolvida como resultado obtido da implementação da arquitetura do Robust OBC.

Figura 44 – Placa de circuito impresso do Robust OBC com medidas de 9 x 9 cm.



Fonte: O autor.

4.3 Testes Funcionais do Robust OBC

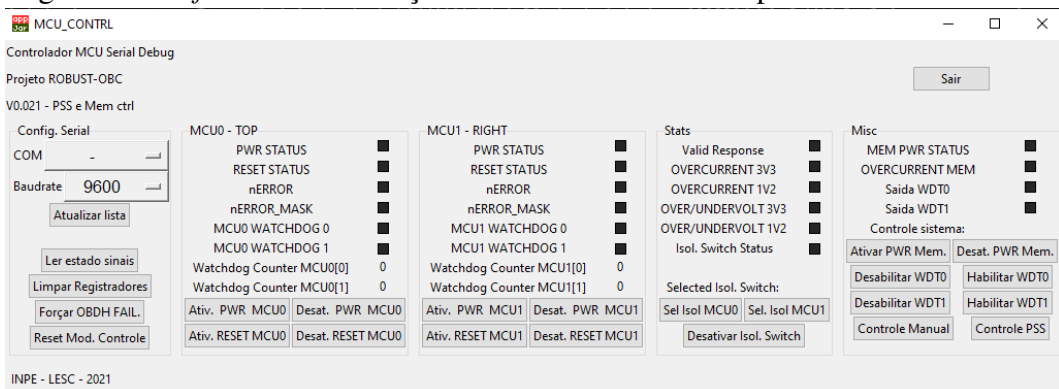
Nesta seção serão demonstrados os procedimentos realizados para execução dos testes de validação e os resultados obtidos de todos os sistemas existentes na arquitetura do Robust OBC. Diversos foram os artefatos utilizados no *setup* de teste para verificação e validação dos sistemas tais quais um *software* de simulação do Sistema de troca de processador, fonte de

alimentação de bancada, carga eletrônica, osciloscópio, multímetro e conversores de comunicação USB-Serial.

4.3.1 Teste do Sistema de Troca de Processador

Para a realização dos testes de funcionamento do PSS, foi desenvolvido um *software* para simular o funcionamento do PSS, no qual é realizado a troca de processadores sem injeção de falhas, ou seja, a troca é feita manualmente por um usuário através do *software*. Neste mesmo *software*, existe a forma de realizar o teste real do PSS. Esse teste acontece quando o *software* é colocado no modo automático fazendo com que o PSS implementado na FPGA atue de maneira conforme idealizado. A comunicação entre o *software* e a FPGA é feita através de uma placa com interface USB/Serial isolada também implementada em tempo de desenvolvimento do doutorado. A tela do *software* de simulação do Sistema de troca de processador é ilustrada na figura 45.

Figura 45 – *Software* de simulação do Sistema de troca de processador.



Fonte: O autor.

Neste *software*, o usuário é capaz de escolher a porta COM serial que irá se comunicar com a FPGA e ainda selecionar a velocidade da comunicação. Outra seleção do usuário é o modo de operação do *software*, se manual ou automático. Existem também, indicadores que mostram a ativação (verde) ou não (vermelho) dos sinais monitorados como processador ativo, *reset*, *nERROR*, *nERROR MASK*, *WATCHDOG*, sobrecorrente, sobre/sub tensão, etc.

No modo manual, o usuário faz todo o procedimento de troca de processador clicando nos botões pertinentes ao processador que deseja-se selecionar. Para a troca, é necessário clicar no botão de *reset*, em seguida o botão de habilitar a alimentação do processador e, por fim, apertar no botão de selecionar as chaves de IO do processador a ser escolhido. Caso queira voltar para o processador anterior, deverá primeiramente desabilitar os botões pertinentes ao processador que está em uso. Outro controle manual é o de OBDH FAIL, que é um botão que

faz com que a placa seja reiniciada simulando uma falha no OBC.

Já no modo automático acontece a simulação real do funcionamento do PSS. Ao clicar no botão "Controle PSS", o sistema funciona conforme sua concepção no qual foram implementados dois *firmwares* distintos e embarcados um em cada processador com objetivo de injetar falhas no sistema para que aconteça a troca de processador. O primeiro *firmware*, embarcado no processador 0, simula um mal funcionamento do processador no qual o *watchdog* deixa de enviar o sinal de vida para o PSS. O segundo *firmware*, embarcado no processador 1, simula uma falha interna no processador (como no caso de um SEU) que o próprio processador não foi capaz de corrigir e assim acionou seu sinal nativo nERROR para que o PSS realizasse a troca de processador. Com isso, o PSS fica realizando o chaveamento entre os processadores e o *software* faz a contagem desse chaveamento para que seja verificada a quantidade de troca em um determinado período de tempo.

Para a realização do teste foi implementado na FPGA um circuito combinacional na linguagem *Verilog* que fosse capaz de realizar o chaveamento dos processadores baseado nos sinais de entrada estipulados na subseção 3.1.1.

Na ocorrência de alguma dessas condições o processador ativo é interrompido e retirado sua alimentação, e o controle do sistema passa para o processador secundário que possui as mesmas habilidades. Supondo que o processador ativo seja o processador 0 (MCU0), o procedimento de reinício e troca de processadores se dará da seguinte forma:

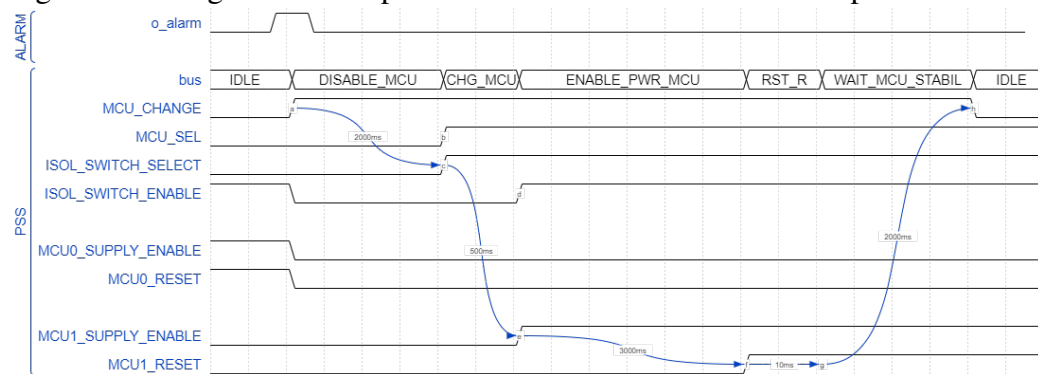
1. Atribuição de nível lógico baixo ao sinal "mcu reset" para ambos os processadores;
2. Atribuição de nível lógico baixo ao sinal "mcu supply enable" para ambos os processadores;
3. Atribuição de nível lógico baixo ao sinal "isol switch enable";
4. Atribuição de nível lógico alto ao sinal "mcu change";
5. Espera do tempo definido pelo parâmetro "START UP WAIT TIME2";
6. Trocar o nível lógico do sinal "isol switch select" para a MCU1;
7. Espera do tempo definido pelo parâmetro "START UP WAIT TIME1";
8. Atribuição de nível lógico alto ao sinal "isol switch enable";
9. Atribuição de nível lógico alto ao sinal "mcu supply enable[1]";
10. Espera do tempo definido pelo parâmetro "START UP WAIT TIME4";
11. Atribuição de nível lógico alto ao sinal "mcu reset[1]";

12. Espera do tempo definido pelo parâmetro “START UP WAIT TIME3”;
13. Atribuição de nível lógico baixo ao sinal “mcu change”.

O MCU CHANGE é um sinal que indica o evento de troca dos processadores, sendo desativado ao final do processo. Este sinal pode ser utilizado por outros módulos para verificar se os processadores ainda estão funcionais.

Durante o processo de troca de processadores há um período de espera entre as mudanças de estado, definido pelos parâmetros INIT MCU WAIT TIME1, INIT MCU WAIT TIME2, INIT MCU WAIT TIME3 e INIT MCU WAIT TIME4, sendo escolhido os valores de 10 ms, 500 ms, 2000 ms e 3000 ms representado no diagrama da figura 46.

Figura 46 – Diagrama de tempo dos sinais do Sistema de troca de processador.



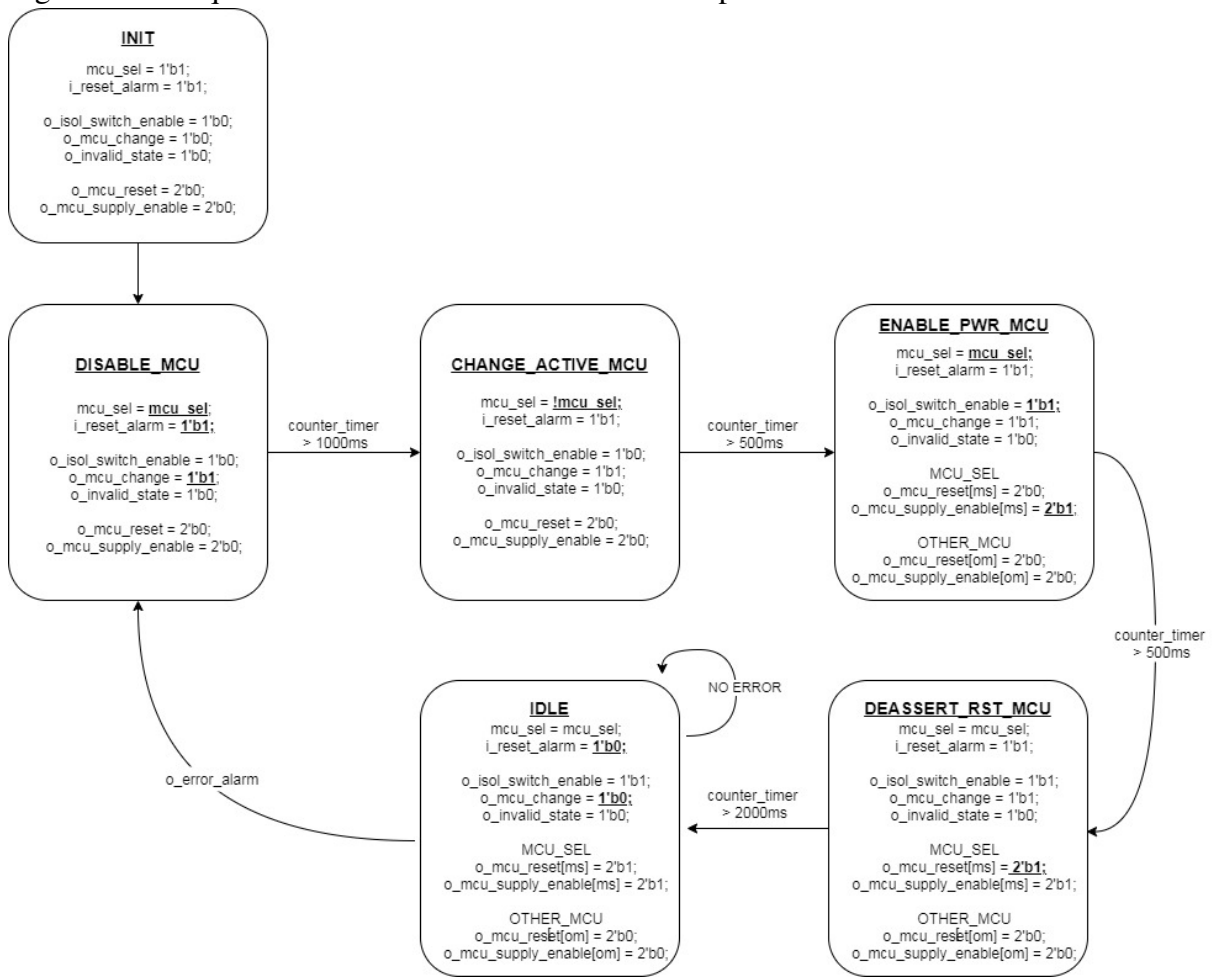
Fonte: O autor.

Logo, o mapa de transições da máquina de estados do Sistema de troca de processador pode ser visto na figura 47.

Outro módulo desenvolvido e pertinente ao Sistema de troca de processador é o módulo de contagem de falhas. Esse módulo tem objetivo de estipular e realizar a contagem da quantidade de troca consecutivas entre os processadores. Caso haja uma determinada quantidade de troca de processador consecutivas em um período de tempo, o módulo entenderá que haverá a necessidade de realizar a troca de placa. Seu funcionamento acontece da seguinte maneira:

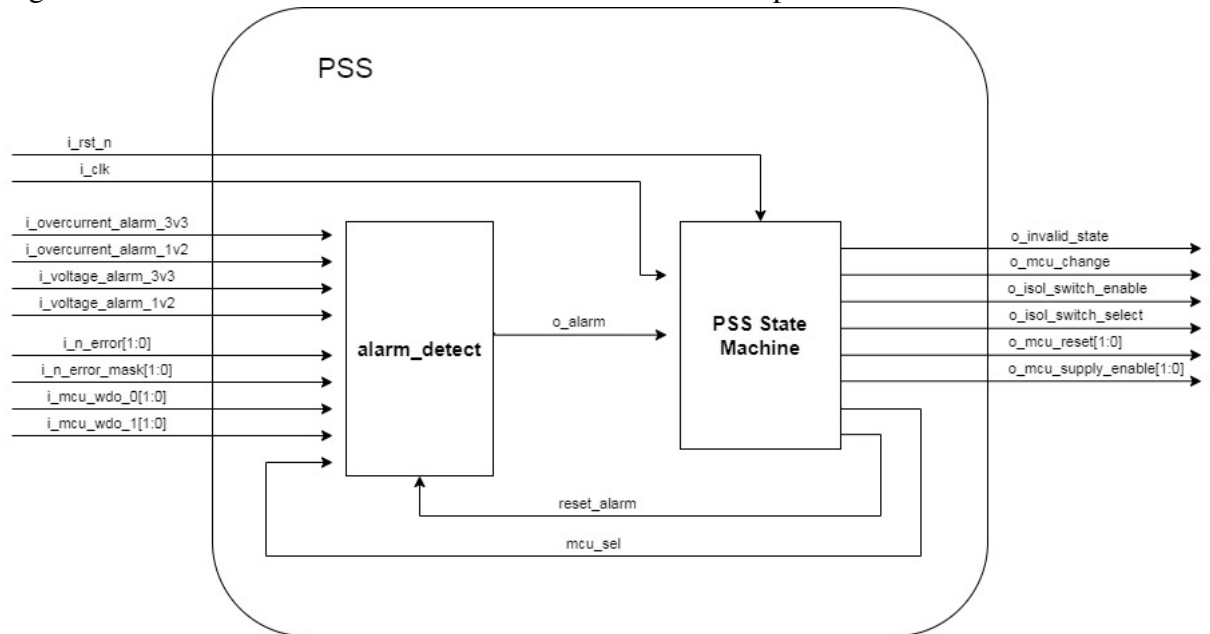
1. O PSS envia o sinal MCU CHANGE toda vez que há troca de processador. Na ocorrência desse evento, o módulo incrementa uma variável interna que representa a quantidade de trocas;
2. Se o processador que assumiu controle do OBC estiver funcionando adequadamente, o sistema deve seguir sem reinicializações. Neste caso, se não forem observados no período especificado no parâmetro THRESHOLD TIME, a variável deve ser decrementada;

Figura 47 – Máquina de estados do Sistema de troca de processador.



Fonte: O autor.

Figura 48 – Sinais de entrada e saída do Sistema de troca de processador.



Fonte: O autor.

3. Entretanto, se o PSS continuar sendo ativado para realizar trocas, isso indica que os processadores estão com problemas permanentes. Neste caso, um limite é estipulado no parâmetro THRESHOLD COUNT, e quando a variável interna ultrapassar este valor, um sinal de falha deve ser gerado.

O sinal de falha deve ser enviado ao processador do Sistema de troca de placa, que deve decidir se realizará a troca do OBC ativo, ou o reinício das fontes do sistema.

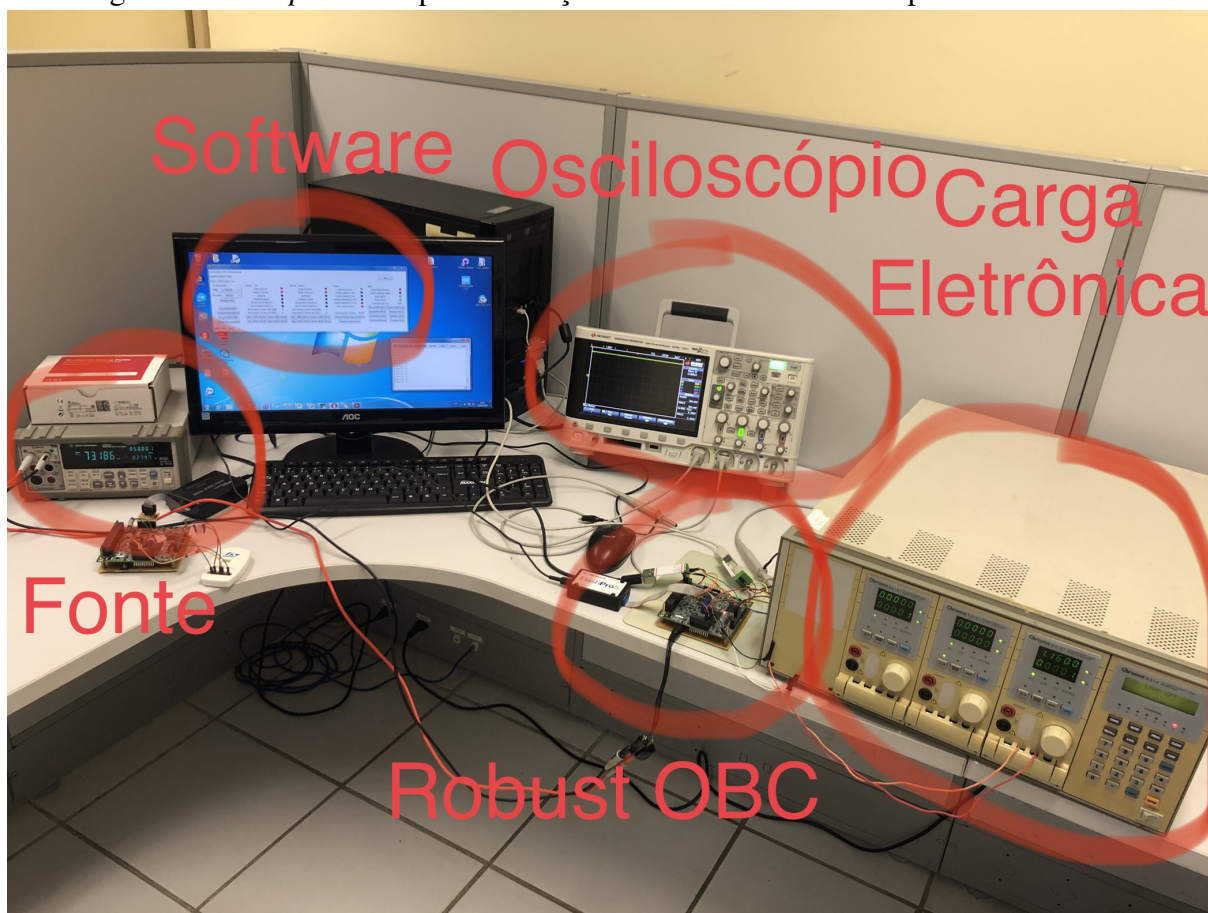
Devido ao longo tempo especificado na variável THRESHOLD TIME, foi utilizado o sinal de *clock Real Time Clock* (RTC) dedicado do sistema, para diminuir o gasto dinâmico de energia e reduzir a demanda de registradores na FPGA.

O Sistema de troca de processador embarcado na FPGA consumiu 154 *LookUp Table* (LUT) de 4 bits e 42 Flip Flop do tipo D (DFF). Esse tamanho se deu por causa do uso de um contador de 28 bits para calcular as transições de tempo dado que está sendo utilizado a frequência de 50 Mhz. Essa quantidade de LUTs e DFFs ainda pode ser reduzida caso a frequência de operação também seja reduzida, pois como o período é maior, menor será a contagem e menor será o circuito associado. A FPGA utilizada possui um total de 12084 LUTs e 12084 DFFs.

Com isso, foi realizado um *setup* de teste para validação do sistema concebido. Esse *setup* consiste em um Robust OBC conectado, via uma placa USB/Serial, no computador que possui o *software* de simulação do Sistema de troca de processador. Uma fonte de alimentação é ligada ao Robust OBC com o intuito de prover energia para a mesma e também para realizar a verificação do consumo da placa para as mais variadas situações de teste. Ainda conectado ao Robust OBC, mais precisamente nos resistores *shunts* que são utilizados para a medição de corrente nos componentes críticos do OBC, uma carga eletrônica modelo 63101 do fabricante Chroma foi utilizada para simular um aumento de corrente nesses componentes como forma de detectar um efeito do tipo SEL. Este teste de SEL, pelo fato de ser uma falha de *hardware*, não está contemplado no *software* e sua intervenção foi feita manualmente. A figura 49 demonstra o *setup* de teste realizado.

Ao energizar o Robust OBC, o processador externo do Sistema de troca de placa (que possui o controle do Gerenciador de falhas juntamente com o processador interna da FPGA) verifica se existem outros OBCs conectados a ele via interface CAN. Nessa situação o consumo de corrente do Robust OBC é de 59 mA. Após essa verificação, o processador externo libera as fontes de alimentação redundantes do Robust OBC e energiza todo o sistema consumindo uma corrente elétrica de 81 mA. Nesse momento a placa fica em estado de espera aguardando algum

Figura 49 – *Setup* de teste para validação do Sistema de troca de processador.



Fonte: O autor.

comando do *software* de simulação de troca de processador.

O teste de validação do Sistema de troca de processador é feito no modo automático clicando no botão "Controle PSS". O processador 0 opera normalmente com seu sinal de vida sendo enviado pelo *watchdog*. Isso ele faz até a contagem de vinte sinais de vida quando, então, ele deixa de enviar este sinal para o PSS que faz todo o processo de troca de processador. Enquanto o processador 0 opera normalmente, o consumo de corrente é de 238 mA. Em seguida, o processador 1 assume o controle e sua forma de operação com injeção de erro é no envio do sinal de nERROR. A operação acontece da seguinte maneira: o processador 1 envia a cada 5 segundos um sinal de nERROR juntamente com o sinal de nERROR MASK, com o objetivo de mascarar o sinal de nERROR de inicialização do processador. Isso ele faz 5 vezes, quando na sexta vez o processador não envia o sinal de nERROR MASK fazendo com que o sinal de nERROR chegue ao PSS como uma falha e inicie todo o processo de troca de processador. O consumo de corrente elétrica do Robust OBC na operação do processador 1 é de 230 mA. O tempo de troca entre os processadores é de 5,5 segundos.

4.3.2 *Teste do Sistema de Troca de Placa*

Para a realização do teste de validação do Sistema de troca de placa foi utilizado o *setup* de teste ilustrado na figura 53. Neste *setup* foram utilizadas duas placas do Robust OBC, dois conversores USB/Serial para comunicação das duas FPGAs e duas instâncias do *software* de simulação do Sistema de troca do processador que se comunicará com as FPGAs através dos conversores citados. O uso da carga eletrônica, dado que a simulação de troca de placa será feita através dos botões "OBC FAIL" e "WATCHDOG FPGA" existentes no *software* de simulação do Sistema de troca de processador, não terá necessidade. O sinal OBC FAIL é a simulação de falhas permanente nos processadores e nas memórias SRAM e o sinal WATCHDOG FPGA é a simulação de falha transitória ou permanente da FPGA.

Houve a necessidade, para o teste do BSS, do desenvolvimento de dois *firmwares* nos dois processadores desse sistema para implementar a idéia concebida na subseção 3.1.3.

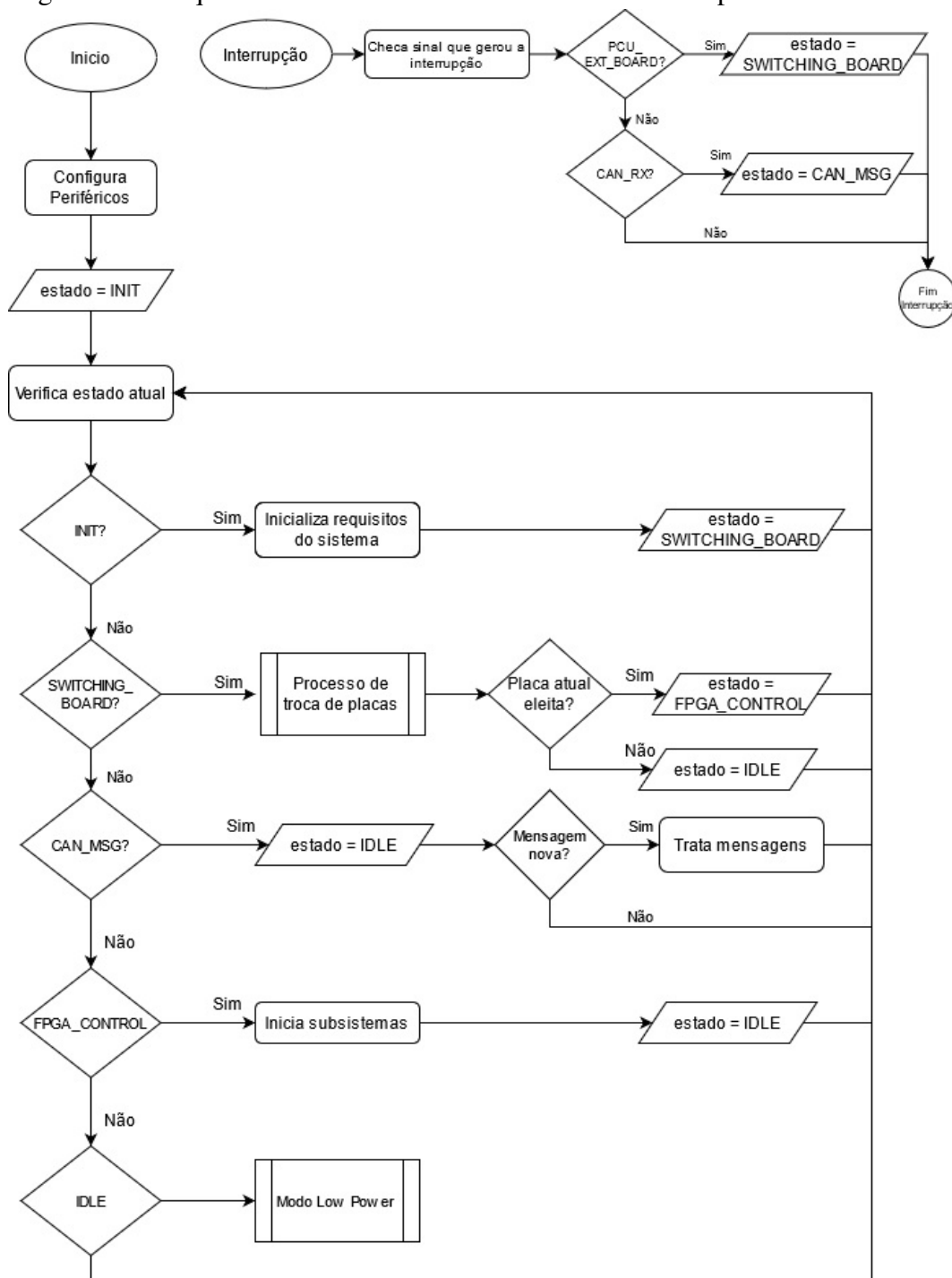
Inicialmente, o processador externo se certifica que os circuitos eletrônicos da sua placa estão desligados, controlando o sinal BSS ENABLE BOARD e depois envia uma mensagem no barramento CAN, solicitando o voto de todas as placas no barramento. De modo geral, o funcionamento do BSS é feito através do barramento de comunicação I/O BSS que contém sinais de estado FM ENABLE BOARD de até três computadores de bordo. Esses sinais são lidos pelos processadores presentes na placa (representados na figura 37 pelos sinais BSS EXT BOARD1 e BSS EX BOARD2) e uma interrupção é gerada ao detectar que esses sinais foram para nível lógico baixo, indicando que o funcionamento do OBC ativo foi interrompido e que o sistema deve iniciar a troca de placas. O funcionamento do *firmware* do processador externo é visto na figura 50.

Com isso, ao receber a solicitação, o Sistema de troca de placa do outro computador de bordo responde com a mensagem do tipo OBC NPLEX SET NEXT BOARD, indicando qual OBC deveria ser o próximo a ser ativado. Dessa forma, todas as placas enviam e recebem mensagens do tipo OBC NPLEX SET NEXT BOARD e OBC NPLEX GET NEXT BOARD.

Após essa fase de votação, o sistema armazena quem participou da votação e a quantidade de votos em cada OBC. Assim, o computador de bordo que será ativado, será o que tem mais votos e que participou da votação. Caso o computador de bordo mais votado não participe da votação, o sistema escolhe o OBC que possui a maior prioridade e que participou da votação. A prioridade do sistema é dada por OBC 1 > OBC 2 > OBC 3.

Por fim, o Sistema de troca de placa do OBC eleito inicia os circuitos eletrônicos da

Figura 50 – Máquina de estado do funcionamento do BSS no processador externo.

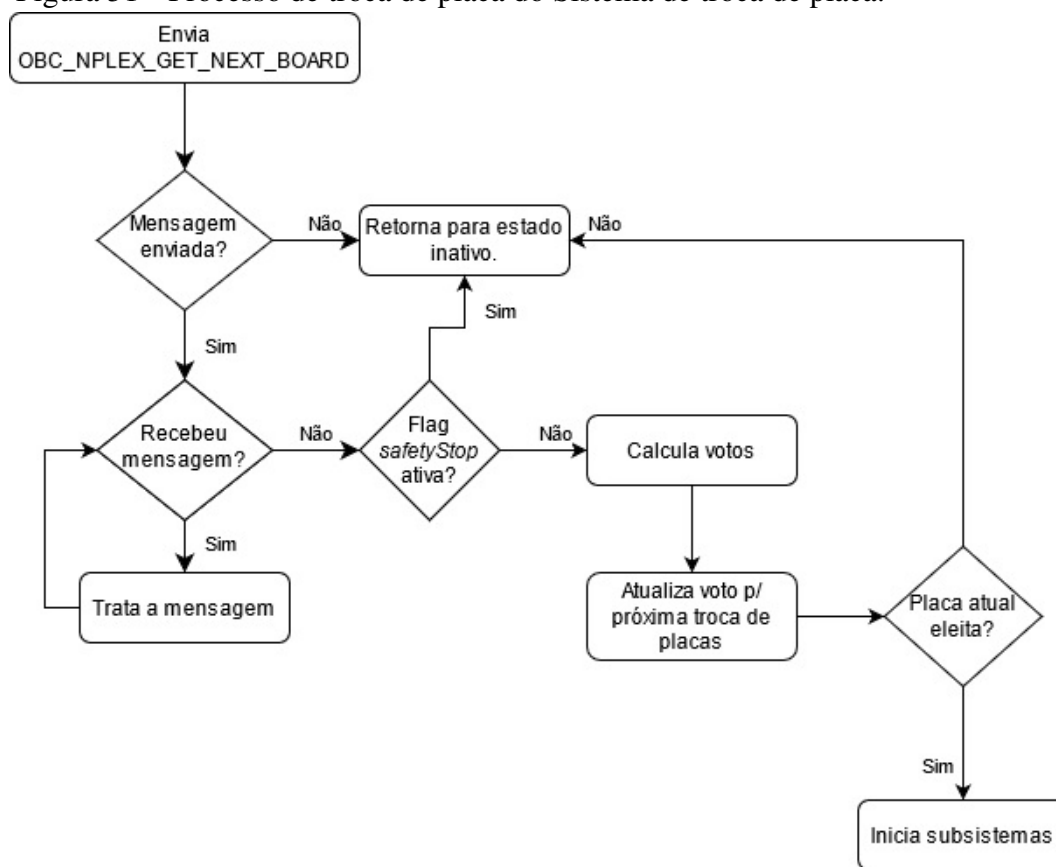


Fonte: O autor.

sua placa e passa o controle para o processador interno da FPGA. Já os outros computadores de bordo, retornam ao estado inativo, monitorando os sinais BSS EXT BOARD1 e BSS EXT BOARD2 para a próxima troca de placas. Uma representação desse processo pode ser vista na figura 51.

O BSS deve garantir que apenas um OBC seja ativado por vez. Para isso, a mensagem

Figura 51 – Processo de troca de placa do Sistema de troca de placa.



Fonte: O autor.

do tipo OBC NPLEX BOARD ACTIVE é enviada tanto pelo BSS de um OBC inativo como pelo BSS no processador interno do OBC ativo. Dessa forma, as falhas críticas que levem apenas um OBC inativo ou os dois reiniciarem, não impactam no OBC ativo.

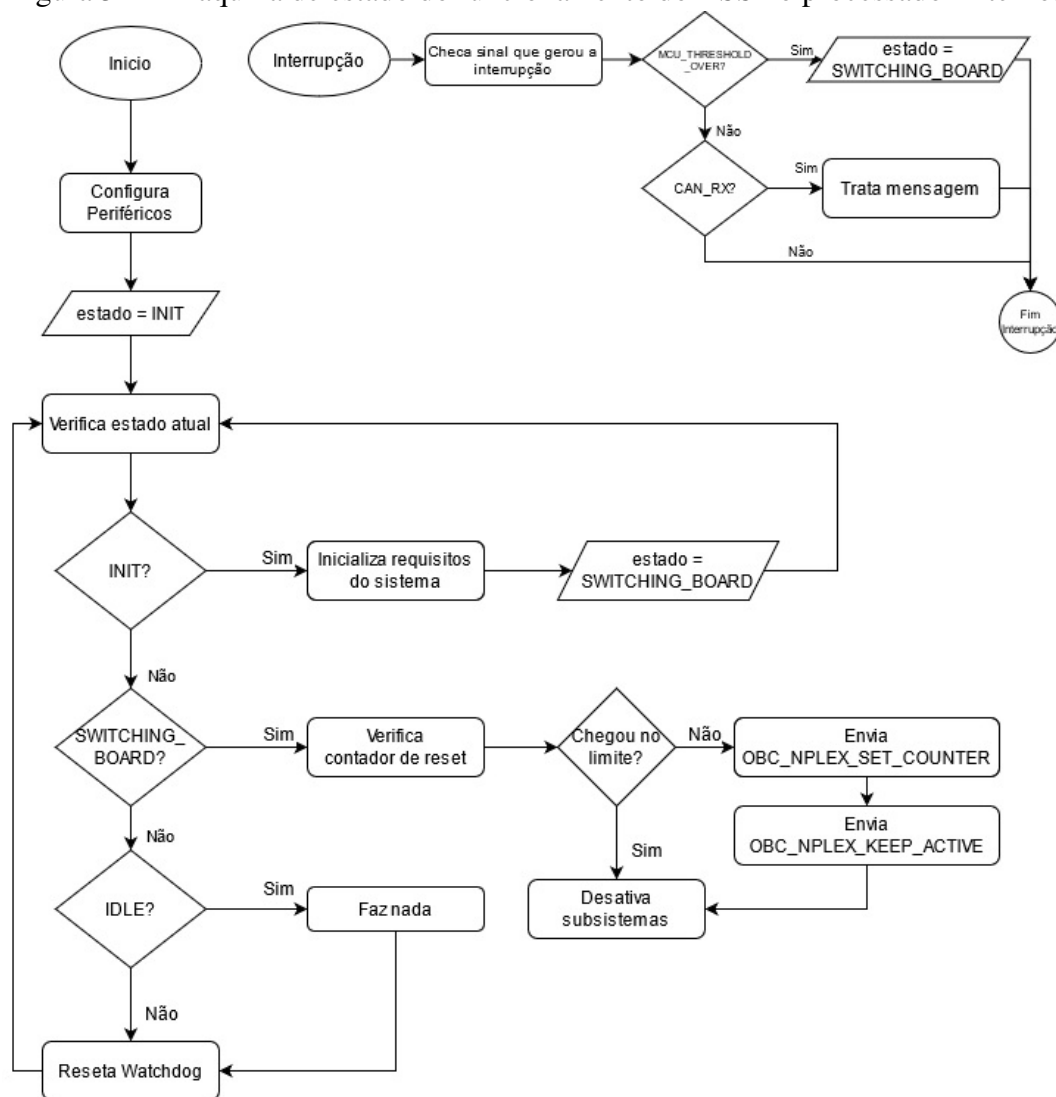
A outra redundância se dá nos próprios sinais BSS EXT BOARD1 e BSS EXT BOARD2, caso algum esteja em nível lógico alto, o OBC eleito não deve iniciar os circuitos eletrônicos, pois isso significa que algum OBC já está ativo.

O processador interno possui acesso aos mesmos sinais do processador externo, além de poder se comunicar com o módulo Gerenciador de *Watchdog* através de algumas *flags* de erro e reiniciar o processador externo. Portanto, o processador interno é responsável por monitorar os sinais BSS EXT BOARD1, BSS EXT BOARD2, para certificar que nenhum outro OBC está ativo, e o sinal MCU THRESHOLD OVER que indica a necessidade de reinicialização do sistema/troca de placa.

A reinicialização através do sinal MCU THRESHOLD OVER não indica diretamente que outra placa deverá ser ativada, para isso, o sistema implementa um contador de *reset* distribuído. Isso significa que todas as placas possuem uma cópia do contador, que é atualizado

através da mensagem do tipo OBC NPLEX SET COUNTER, enviada momentos antes da placa ativa passar o controle para o processador externo e ser reinicializada. Além disso, é enviada a mensagem do tipo OBC NPLEX KEEP ACTIVE, para que os votos sejam na placa atual, e ela seja escolhida novamente. Esse processo vai ser repetido algumas vezes, determinado em *firmware* através do parâmetro MAX RESET THRESHOLD. A representação da máquina de estado do *firmware* do processador interno pode ser vista na figura 52.

Figura 52 – Máquina de estado do funcionamento do BSS no processador interno.



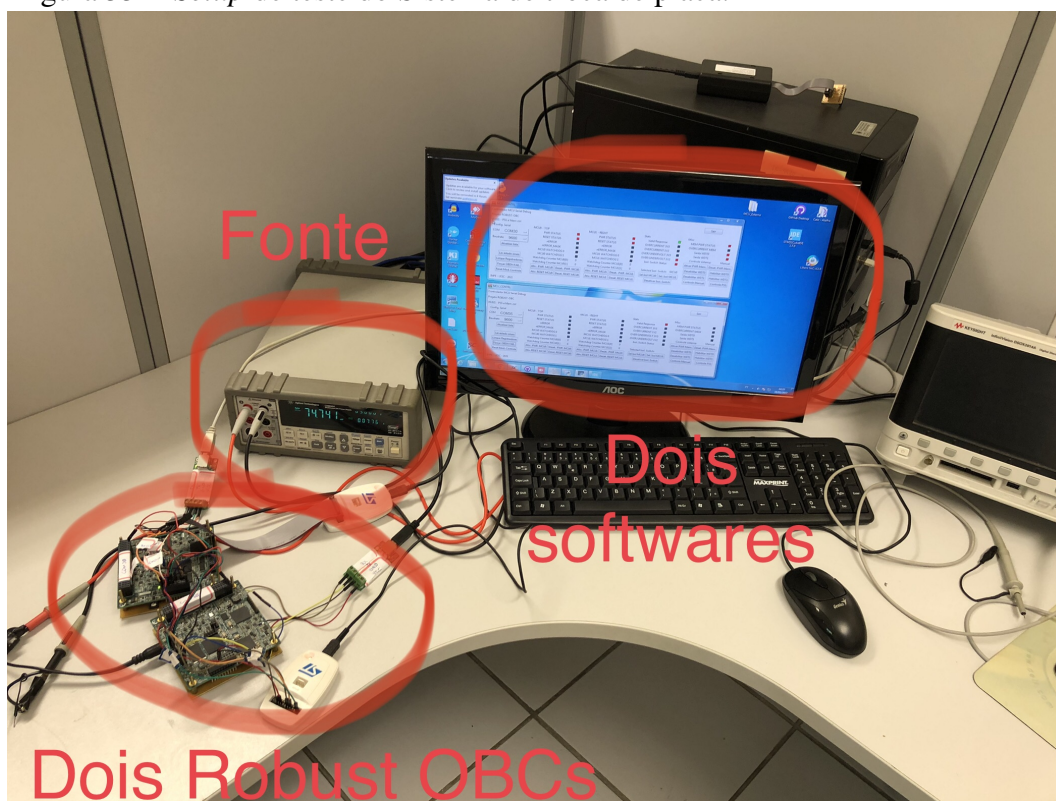
Fonte: O autor.

Para a realização do teste de validação foram utilizadas somente duas placas do Robust OBC, pois só haviam componentes eletrônicos para a montagem de somente as duas placas citadas.

Ao energizar as fontes redundantes do Sistema de troca de placa dos dois Robust OBCs através da fonte de alimentação do fabricante Keysight modelo U3606A mostrada na

figura 53, o consumo de ambas é de 77 mA no Robust OBC 1 e 13 mA no Robust OBC 2 dado que o Robust OBC 1 é selecionado como ativo através de votação feita nos processadores externos de ambas as placas e o Robust OBC 2 está inativo. Feito isso, o Robust OBC 1 está ativo e ao ligar o modo de chaveamento entre os processadores no *software* de simulação do Sistema de troca de processador seu consumo é de 250 mA enquanto o Robust OBC 2 está inativo e consumindo 13 mA. Logo, o *software* de simulação do Sistema de troca de processador está apto ao uso para realizar o chaveamento de placa e para simular uma falha catastrófica no Robust OBC 1, é pressionado o botão "OBC FAIL" para simulação de falhas permanentes dando a entender que algum componente crítico desse OBC está danificado.

Figura 53 – *Setup* de teste do Sistema de troca de placa.



Fonte: O autor.

Ao ativar a placa Robust OBC 2 e desativar o Robust OBC 1, seus consumos ficam em 97 mA e 10 mA respectivamente e ao acionar o modo de PSS no *software* de simulação do Sistema de troca de processador, o consumo do Robust OBC 2 vai para 268 mA e o Robust OBC 1 continua em 10 mA. Para realizar a simulação de uma falha transitória, é necessário apertar o botão "WATCHDOG FPGA" que simula uma falha transitória na FPGA ou em seus módulos internos. Com isso, todo o processo de troca de placa é feito novamente, passando o controle do nano satélite para o Robust OBC 1. O tempo de troca entre as placas é de 3,5 segundos.

4.3.3 Teste do Sistema de Comunicação do Processador e da Memória

Esse sistema foi testado de duas maneiras. A primeira foi usando o *software* de simulação do Sistema de troca de processador em conjunto com a carga eletrônica conforme a figura 49 e a segunda foi embarcando na FPGA os módulos de comunicação entre o processador e a memória incluindo o módulo contendo os EDACs.

No primeiro teste foi conectada a carga eletrônica no resistor *shunt* na fonte de 2,5 V que alimenta as memórias SRAM para controlar a corrente elétrica que passa sobre esse resistor. Esse tipo de teste é para simulação de um SEL sobre as memórias. Em seguida, foi apertado o botão "Ativar PWR mem." para ativar a alimentação das memórias SRAM e ao verificar que a corrente está no limite (395 mA) do estipulado (400 mA) a FPGA, através do Gerenciador de falhas, desliga a alimentação das memórias através das chaves redundantes, eliminando a possibilidade de queima das memórias através de SEL.

O segundo teste que diz respeito a comunicação entre os componentes processador, FPGA e memória SRAM foi testado por partes para eliminação de problemas pertinentes a comunicação integrada desses dispositivos.

Primeiramente foi testado a comunicação entre a FPGA e as memórias SRAM. Nesse teste foi necessário desenvolver um *firmware* para ser embarcado no processador interno da FPGA para escrever e ler dados nas regiões de memória com objetivo de validar o roteamento dos barramentos e a montagem dos componentes.

Para validar o barramento de dados, foi utilizado o método para testar o barramento de um bit de cada vez. O barramento de dados passa no teste se cada bit de dados puder ser definido como 0 e 1, independentemente dos outros bits de dados. Com isso, é necessário escrever o primeiro valor de dados na memória, verificando-o e lendo-o de volta, depois escrever o segundo valor, verificar e assim por diante. Quando chegar ao final da memória, o teste estará concluído.

Já o barramento de endereço, o teste é para verificar se existe sobreposição em duas regiões de memórias, logo deve-se primeiro escrever algum valor de dados inicial em cada local de potência de dois no dispositivo. Em seguida, escrever um novo valor (uma cópia invertida do valor inicial é uma boa escolha) no local de teste e verificar se o valor de dados inicial ainda está armazenado em todos os outros locais de potência de dois. Se não for encontrado um local, diferente daquele que acabou de ser escrito, que contém o novo valor de dados, haverá um problema com o bit de endereço atual. Se nenhuma sobreposição for encontrada, é necessário

repetir o procedimento para cada um dos locais restantes.

Esses testes foram realizados nas duas placas e foi verificado que na placa Robust OBC 2 existe um curto circuito em dois sinais de dados acarretando em uma palavra menor de armazenamento e em dois sinais de endereço acarretando na diminuição no espaço de memória. A placa Robust OBC 1 passou no teste e está em pleno funcionamento.

Feito isso, foi realizado o teste da comunicação entre o processador e a FPGA. Para a realização desse teste foi instanciado na FPGA uma memória com objetivo de aplicar leituras e escritas na mesma. Com isso, foi desenvolvido um *firmware* no processador TMS570 que exercesse essa função. Primeiramente, a leitura dos dados escritos nessa memória foram feitos através do *software* de depuração da FPGA no computador pessoal. Depois de verificado este bom funcionamento, foi feita a leitura através do próprio processador. Nesse caso, o processador escrevia em uma região de memória estipulada pelo barramento de endereço e em seguida lia verificando se os dados eram o mesmo que haviam sido escritos, validando assim a comunicação entre ele e a FPGA.

Para efetuar a comunicação completa, foi implementado na FPGA alguns blocos que juntos realizam essa comunicação, são eles:

1. Bloco SDRAM2SRAM;
2. Bloco Data Split;
3. Bloco ECCs Bank;
4. Bloco FPGA Config.

O bloco SDRAM2SRAM é o responsável por receber os dados do controlador de memória do processador e emular o funcionamento de uma memória SDRAM, do inglês , para tornar possível a comunicação com as memórias SRAM. Desta forma, a ideia do bloco é criar uma interface entre o processador, que se comunica com a memória externa e o sistema desenvolvido na FPGA, onde as memórias SRAM estão inseridas. Partindo deste princípio, o bloco possui duas operações mais críticas que são: as conversões das escritas e leituras em SDRAM para SRAM.

O bloco Data Split é responsável por transformar um sinal de entrada e saída (DATA) em dois sinais, um de entrada (DATA WRITE) e outro de saída (DATA READ). Quando o sinal OE WE indicar que está havendo uma leitura, o bloco deve atribuir ao sinal de saída DATA READ o conteúdo do sinal DATA, caso contrário, o bloco deve atribuir ao sinal DATA o valor do sinal de Entrada DATA WRITE.

O bloco ECCs Bank é responsável por realizar a codificação e decodificação, usando um entre três possíveis EDACs. Foram utilizados os códigos DAEC, TAEC e QUAEC que, respectivamente, podem corrigir até dois, até três e até quatro erros em rajada. Um erro em rajada é definido como um erro múltiplo que abrange L bits em uma palavra, isto é, um grupo de bits contíguos onde, pelo menos, o primeiro e o último bits estão com erro. L é definido como o comprimento do erro em rajada.

O bloco FPGA Config é responsável por selecionar qual EDAC será utilizado de acordo com o padrão de erro detectado por esse bloco.

Após a implementação desses módulos, os mesmos foram embarcados na FPGA nos quais cada um consumiu uma área conforme abaixo e fez com que a placa do Robust OBC obtivesse um consumo energético de 125 mA.

1. Bloco SDRAM2SRAM = 132 LUT e 75 DFFs;
2. Bloco Data Split = 47 LUTs;
3. Bloco ECCs Bank = 777 LUTs;
4. Bloco FPGA Config = 15 LUTs e 46 DFFs.

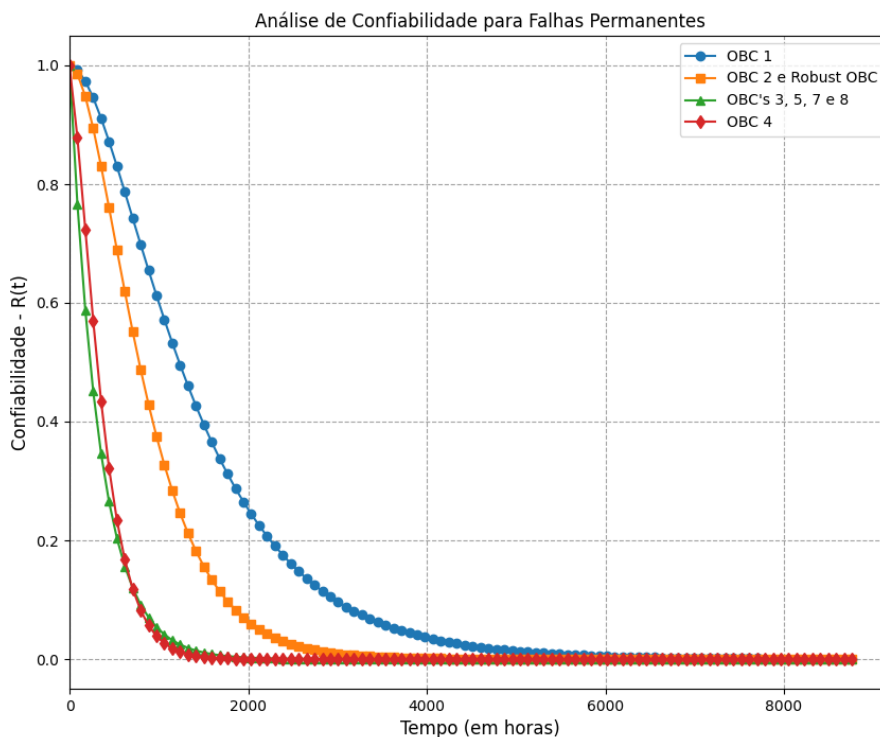
4.3.4 Resultado e Comparação de Confiabilidade

Para a geração dos gráficos de confiabilidade, foram utilizados os valores de referência dos parâmetros λ e μ do trabalho de (FAYYAZ; VLADIMIROVA, 2014). A taxa de falha utilizada foi $\lambda = 1 \times 10^{-3}$ e a taxa de recuperação foi $\mu = 1 \times 10^{-5}$. A análise foi feita com o tempo de duração de um ano.

A primeira análise foi feita para a confiabilidade em falhas permanentes. Como dito anteriormente, o difícil acesso aos valores de confiabilidade, taxas de falhas, de recuperação dos componentes eletrônicos como processador, FPGA e memórias fez com que fossem utilizados os valores de λ e μ do trabalho (FAYYAZ; VLADIMIROVA, 2014) igualmente para todos os componentes descritos em cada arquitetura analisada, assim como a estipulação da confiabilidade do votador igual a 1. Além disso, para o caso do Robust OBC, que foram utilizados componentes críticos dos fabricantes Texas Instruments e Microsemi que fornecem os valores de taxa de falhas, como por exemplo $2,41 \times 10^{-9}$ e $2,71 \times 10^{-9}$ respectivamente, não foram utilizados na análise. Esses valores inseridos na análise fariam com que a confiabilidade não baixasse durante o período de tempo analisado. Isso acontece porque os testes que originaram esses valores foram feitos para o âmbito terrestre. Outra referência para a taxa de falha em memórias foi encontrada

em (VARGAS, 1995) para o caso em aplicações espaciais, porém não foi utilizada nesta tese por que, para o tempo de um ano, não há decréscimo de confiabilidade. A figura 54 ilustra a comparação entre o Robust OBC e os demais OBCs descritos nssa tese.

Figura 54 – Gráfico de comparação das confiabilidades para falhas permanentes dos OBCs.



Fonte: O autor.

Como mostrado no gráfico da figura 54, o OBC 1 é o que possui a maior confiabilidade ao longo do tempo. Isso se dá pelo fato de que sua arquitetura é a que possui a menor quantidade de elementos que a integram, e essa integração ainda é em forma de redundância, justificando seu maior valor de confiabilidade. Como desvantagem, a limitação do seu sistema de computação faz com que esse OBC seja incapaz de produzir aplicações de alto desempenho.

O Robust OBC e OBC 2 possuem a mesma confiabilidade ao longo do tempo dado que seus modelos de arquitetura são os mesmos para falhas permanentes. Esses OBCs são, atrás somente do OBC 1, os que possuem a maior confiabilidade com uma arquitetura de sistema de computação completa capaz de produzir aplicações de alto desempenho. Além disso, esses OBCs ainda possuem componentes com características de tolerância a falhas em sua construção como o processador TMS570 e possuem componentes comprovadamente robustos, através de testes

realizados e com experiência em missões espaciais como a FPGA no Robust OBC e o supervisor no OBC 2. As diferenças entre o Robust OBC e o OBC 2 estão nas técnicas de tolerância a falhas utilizadas em ambos. O Robust OBC possui as técnicas de redundância, *watchdog*, circuito de detecção de SEL e EDAC, enquanto o OBC 2 possui somente redundância e *watchdog*.

Já o OBC 4 possui a terceira maior confiabilidade, pois o mesmo possui dois sistemas de redundância TMR para proteção de suas memórias SRAM e Flash. Isso faz com que sua confiabilidade seja inferior aos OBCs do padrão 2 e superior aos OBCs do padrão 3 que não possuem nenhum tipo de redundância.

O que faz a diferença na vida útil dos OBCs do padrão 3 são as técnicas de tolerância a falhas empregadas em cada um deles e o uso ou não de componentes que possuem técnicas de tolerância a falhas em sua construção. Isso é mostrado na Tabela 1.

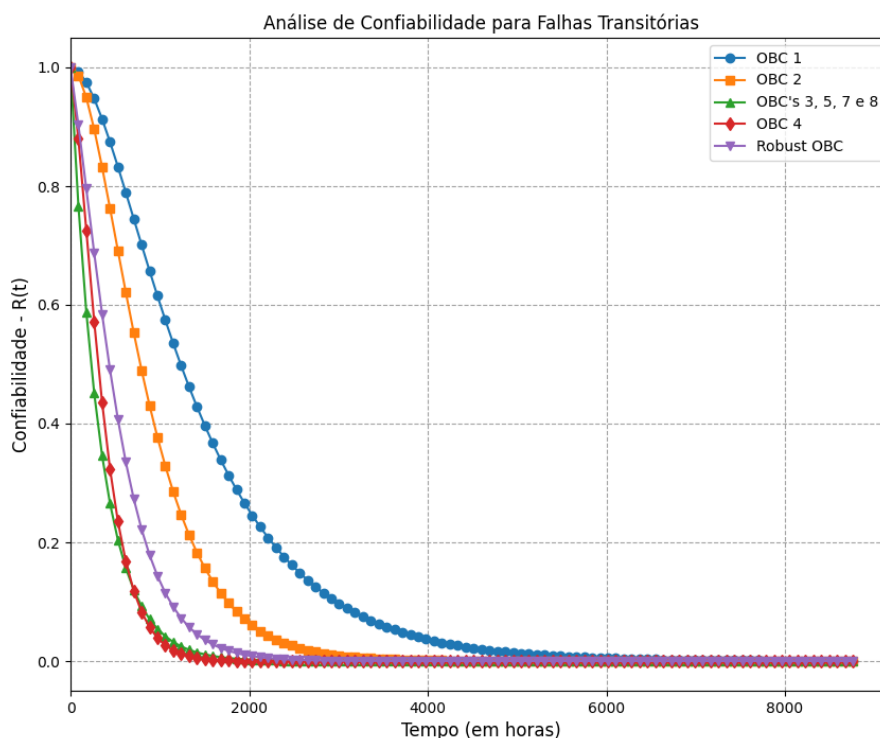
O gráfico das falhas transitórias ilustrado na figura 55 possui sua análise baseada na recuperação de efeitos como SEU. Em uma arquitetura com menos elementos e com uma técnica de tolerância a falhas robusta será a que irá possuir a maior confiabilidade. Como é o caso do OBC 1, que possui a técnica de redundância a frio em seus processadores fazendo com que esse OBC possua a maior confiabilidade para falhas transitórias em detrimento aos demais.

Em segundo lugar vem o OBC 2, que possui a mesma técnica de redundância de processador, porém possui comunicação com memórias aumentando elementos e diminuindo a confiabilidade.

Logo após o OBC 2 vem o Robust OBC que, além de possuir a mesma técnica de redundância de processadores a frio usando a FPGA como supervisor, ainda possui na mesma FPGA códigos corretores de erros para proteção de dados nas memórias SRAM. Isso faz, do ponto de vista de modelagem de arquitetura, com que a FPGA seja vista como dois elementos acarretando na diminuição de confiabilidade. Porém em termos práticos, a utilização de uma FPGA respaldada pelo seu fabricante através de testes de radiação e com sua experiência em missões espaciais, é muito importante seu uso para atribuições de tolerância a falhas como os sistemas propostos nessa tese e implementadas nessa FPGA. Além disso, a FPGA ainda possui proteção contra SEL através de um circuito de detecção para o mesmo.

Em seguida aparece o OBC 4, que através de suas técnicas de TMR, possui a confiabilidade superior aos OBCs do padrão 3 dado que possuir redundância de memórias é mais confiável que utilizar apenas um EDAC como proteção de dados na memórias. No âmbito da relação custo/benefício, o uso do EDAC é mais eficiente, pois possui diversas memórias para

Figura 55 – Gráfico de comparação das confiabilidades para falhas transitórias dos OBCs.



Fonte: O autor.

garantir que não haja erros em seus dados trás alguns efeitos negativos como um maior custo financeiro, energético e de espaço físico.

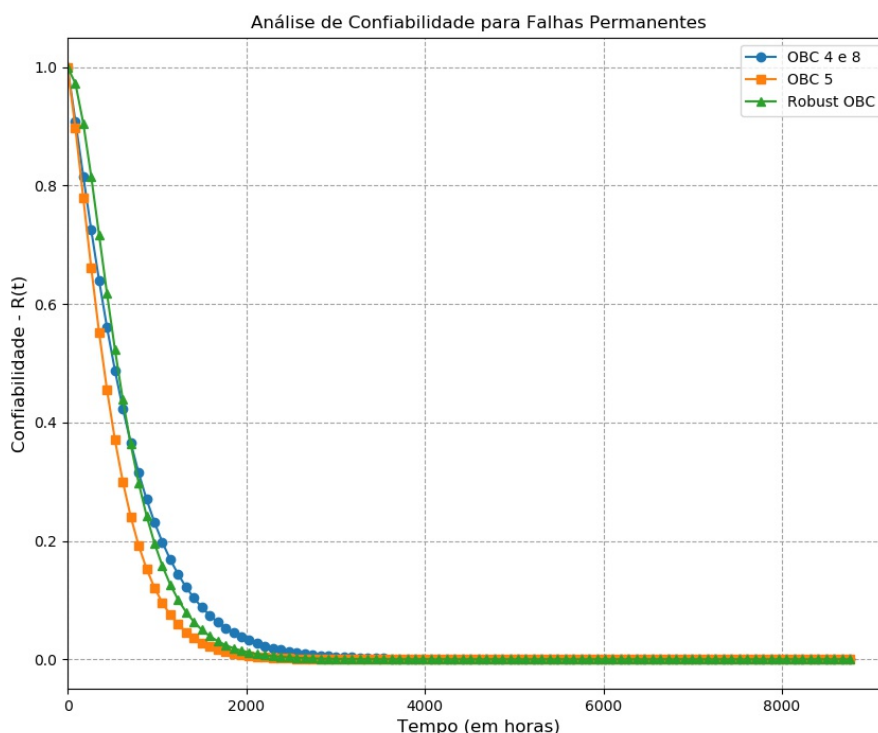
Com relação ao uso de redundância de OBC no nano satélite, somente os OBCs 4, 5, 8 e o Robust OBC possuem a capacidade de realizar essa função. Como dito no capítulo 2, a análise da confiabilidade de redundância será feita apenas para o caso de falhas permanentes dado que somente haverá troca de OBC caso o OBC em uso não funcione mais.

A figura 56 ilustra a confiabilidade no tempo para falhas permanentes da redundância de OBCs. A redundância no votador/supervisor para a realização da troca de placa foi fundamental para que o Robust OBC obtivesse a maior confiabilidade entre o sistema de troca de OBC utilizado por cada OBC.

4.4 Conclusão

Neste capítulo, foi identificado que o método FMEA é uma ferramenta muito importante para análise de falhas de um sistema de computação como o Robust OBC. Seu uso fez

Figura 56 – Gráfico de comparação das confiabilidades para falhas permanentes na redundância de OBCs.



Fonte: O autor.

com que fosse compreendido todo o caminho realizado, através da árvore de falha, das falhas permanentes e transitórias nos componentes críticos utilizados nessa tese. Com isso, tornou-se simples a tomada de decisão para mitigar os problemas pertinentes a essas falhas.

A criação de um *setup* de teste que contemplasse todos os testes de validação das idéias concebidas nas fases de pesquisa e desenvolvimento foi de muita importância para a aquisição de resultados sólidos que demonstrasse o bom funcionamento dos circuitos fazendo com que os mesmos estivessem aptos a serem utilizados em aplicações reais.

O teste do Sistema de troca de processador ocorreu da maneira correta e esperada alinhando a teoria com a prática, e demonstrou que a questão de inserção de variáveis que possam ser alteradas de acordo com a aplicação que o Robust OBC será inserido, faz com que use-se menos ou mais LUTs e DFFs, aumente-se ou diminua-se o tempo de chaveamento de processador e consume-se menos ou mais corrente elétrica.

Assim como o teste anterior, o teste do Sistema de troca de placa também se mostrou dentro do esperado implicando que as idéias concebidas para esse sistema foram bem elaboradas. O consumo de corrente elétrica no chaveamento entre as placas é considerado baixo, assim como

o tempo desse chaveamento.

O teste de validação do Sistema de comunicação entre o processador e a memória aconteceu de quatro formas. A primeira foi o teste de proteção da memória contra SEL, no qual seu funcionamento deu-se de maneira correta. A segunda foi o teste da comunicação entre o processador e a FPGA no qual foi simulado as ações de escrita e leitura em uma memória instanciada na FPGA para a verificação do funcionamento dos barramentos, ocorrendo tudo dentro do esperado. A terceira forma foi o teste da comunicação entre a FPGA e a memória. Esse teste foi feito realizando escritas e leituras na memória para verificação da integridade dos barramentos e foi visto que em uma das placas do Robust OBC existem curtos circuitos em dois sinais no barramento de dados e dois sinais do barramento de endereço. Por fim, a quarta forma de teste foi na implementação dos módulos desses sistemas de comunicação dos testes anteriores juntamente com o módulo de EDACs para serem embarcados na FPGA. Como resultado foi um total de 971 LUTs e 121 DFFs e um consumo corrente elétrica de 125 mA. Não houve um teste de detecção e correção de erros dos EDACs embarcados, pois os mesmos já foram validados em (GRACIA-MORÁN *et al.*, 2018) e para a realização desse teste de forma concreta é necessário um injetor de radiação no qual será realizado em trabalhos futuros.

Ainda neste capítulo, foi verificada e analisada a confiabilidade dos modelos de arquitetura do sistema de computação dos OBCs. A fidedignidade do valor de confiabilidade do sistema de computação está totalmente atrelada ao valor de confiabilidade de cada componente, cuja sua aquisição é de difícil acesso. Foi examinado que um sistema de computação com o menor número de elementos possui uma maior confiabilidade, porém o mesmo é limitado no que diz respeito ao desempenho computacional. Outra percepção vista foi que as técnicas de tolerância a falhas, mesmo sem a posse de uma métrica referente a elas, é um diferencial para aumentar confiabilidade quando os modelos de arquitetura são iguais. Com isso, o Robust OBC se mostrou a melhor opção entre os OBC analisados, pois o mesmo alia um modelo de arquitetura com um grande desempenho computacional e uma alta confiabilidade, tanto para uma unidade quanto para o modelo de redundância de OBCs, com técnicas de tolerância a falhas e com componentes eletrônicos COTS que possuem resistência e imunidade comprovadas contra os efeitos da radiação.

5 CONCLUSÕES E PERSPECTIVAS

Essa tese teve como principal contribuição a pesquisa e desenvolvimento de uma arquitetura de um computador de bordo COTS para nano satélite de padrão CubeSat que fosse robusta, com alto desempenho computacional e de alta confiabilidade capaz de exercer missões espaciais simples ou críticas.

Para isso foi pesquisado, na fronteira do conhecimento, diversos computadores de bordo para realização de um *brainstorm* com intuito de conhecer o que está sendo desenvolvido no meio industrial e na comunidade acadêmica nesse nicho de mercado. Esse processo de escolha foi bastante criterioso realizando uma filtragem necessária, pelo fato de ter sido encontrado uma grande quantidade de OBCs que não se enquadrava nos requisitos impostos para o desenvolvimento do Robust OBC. Além disso, foi pesquisado e estudado diversas técnicas de tolerância a falhas utilizadas em ambiente espacial capazes de combater os efeitos provenientes desse meio para aumentar a confiabilidade e a vida útil do OBC objeto dessa tese.

Após esse estudo, foi projetado alguns sistemas que pudessem tolerar falhas permanentes e transitórias. Sistemas esses que fazem uso de técnicas de tolerância a falhas reconhecidas na comunidade acadêmica, mas com algum diferencial dando um tom de inovação ao que foi implementado nesse computador de bordo.

O resultado disso foi o desenvolvimento de uma placa de circuito impresso de dez camadas distribuídas entre roteamento de sinais e planos de terra e alimentação, com o tamanho estipulado no padrão CubeSat (THE CUBESAT STANDARD, 2021) para a realização dos testes dos sistemas concebidos na arquitetura proposta do Robust OBC.

Os testes dos sistemas projetados foram bem elaborados mostrando que seu funcionamento ocorreu de acordo com o que foi pensado. Os testes foram feitos em laboratório em um ambiente controlado fazendo com que o Robust OBC esteja apto para ser testado em ambiente de simulação espacial.

Foi visto que um *setup* de teste com os equipamentos adequados traz grande credibilidade aos resultados obtidos. Um outro fator importante, que foi um diferencial para a coleta dos dados que fez validar os sistemas projetados, foi o desenvolvimento de um *software* capaz de simular as falhas através de sinais importantes descritos nessa tese.

Um outro resultado que se mostrou imprescindível foi a comparação da confiabilidade do modelo de sistema de computação dos OBCs pesquisados. Foi identificado que os OBCs seguem alguns padrões de arquitetura tendo como diferença entre eles os componentes

escolhidos para implementar essa arquitetura e as técnicas de tolerância a falhas selecionadas para proteção contra os efeitos existentes no ambiente espacial.

Com isso, o Robust OBC se mostrou uma opção bastante tentadora, pois seu padrão de sistema de computação possui capacidade para alto desempenho, possui técnicas de tolerância a falhas que abrange todas as possíveis falhas provenientes dos efeitos da radiação, além de utilizar componentes eletrônicos COTS com características de tolerância a falhas em sua construção e comprovadamente imune a esses efeitos através de testes realizados e com experiência em ambiente espacial.

De acordo com os objetivos estipulados no capítulo 1, esse trabalho alcançou os objetivos com êxito através de resultados sólidos demonstrados ao longo da tese. Ainda observando o capítulo 1 foram levantados alguns questionamentos que suas respostas indicam contribuições bastante relevantes para pesquisadores que planejam se inserir nesta área de pesquisa. Respostas essas que serão respondidas logo abaixo:

1. Uma arquitetura tolerante a falhas utilizando componentes COTS é capaz de obter a mesma confiabilidade que uma outra utilizando componentes Rad-Hard?
Resposta: Sim. Porém a medida que o tempo passa a confiabilidade dos componentes eletrônicos COTS diminui. Para missões de nano satélites de padrão CubeSat é totalmente justificável o uso de uma arquitetura utilizando componentes COTS porque o tempo médio de sua missão é de dois anos, tempo suficiente para o bom funcionamento desses tipos de componentes.
2. Caso o processador sofra uma falha permanente, o que acontecerá com o OBC?
Resposta: Se um processador do Robust OBC falhar permanentemente haverá troca para o processador redundante, caso o segundo falhe também haverá troca de Robust OBC.
3. Não é possível prever qual o tipo de erro lógico irá acontecer na memória SRAM, logo, qual código corretor de erro mais adequado embarcar para mitigar essa falha? Somente um código é suficiente?
Resposta: Para isso, foi utilizado três EDACs que podem corrigir até dois, até três e até quatro erros em rajada e que atuam dinamicamente de acordo com o tipo de erro detectado.
4. Qual a melhor maneira de aumentar a confiabilidade em um sistema de alimentação de energia?

Resposta: Foi verificado que o uso de redundância com reguladores de tensão diferentes que absorvam os efeitos de radiação de maneira diferente é de grande importância para o aumento de confiabilidade nesse sistema.

5. É sabido que os computadores de bordo possuem componentes com maior importância que outros, além disso, como o OBC é de baixo custo, o que invalida o uso da técnica de redundância na maioria dos pontos únicos de falha, o que deverá ser feito caso o computador de bordo perca suas principais funções permanentemente?

Resposta: A utilização de redundância de OBC com até três unidades resolve a questão de pouco espaço para redundância de componente em uma única placa. Ou seja, um Robust OBC possui redundância nos seus sistemas mais importantes e o nano satélite possui um sistema de redundância de até três Robust OBCs aumentando ainda mais a confiabilidade do "cérebro" do satélite.

Como a análise de confiabilidade do Robust OBC foi realizada utilizando valores de taxa de falhas e recuperação disponível na comunidade acadêmica de forma genérica, será necessário um trabalho futuro que possa levantar esses dados afim de otimizar a confiabilidade do Robust OBC.

Outro trabalho futuro que deverá ser realizado é a submissão de testes de radiação no Robust OBC para poder avaliar o funcionamento de todos os sistemas projetados em um ambiente hostil com o intuito de simular de forma mais próxima da realidade o ambiente espacial onde este OBC estará submetido.

REFERÊNCIAS

- AL-ARIAN, S.; GUMUSEL, M. Hptr: Hardware partition in time redundancy technique for fault tolerance. In: **Proceedings IEEE Southeastcon '92**. [S. l.: s. n.], 1992. p. 630–633 vol.2.
- AMEEL D. AMIDEI, K. S. E. R. E. Y. Y. P. C. N. D. A. L. S. B. S. F. M. C. S. L. M. L. J. New small wheel low-voltage power: Design review. In: **CERN**. [S. l.: s. n.], 2015. p. 1–25.
- ANALOG DEVICES. **ADP1755ACPZ-R7 Datasheet**. 2014. Disponível em: https://www.analog.com/media/en/technical-documentation/data-sheets/ADP1754_1755.pdf. Acesso em: 16 dez. 2021.
- BACELO, J. R. C. **Análise do Risco e da Confiabilidade em Sistemas Complexos e Reparáveis**. 2016. Dissertação (Mestrado em Engenharia Eletrônica e Computação) – Centro de Ciências Sociais e Tecnológicas, Programa de Pós-Graduação em Engenharia Eletrônica e Computação, Universidade Católica de Pelotas, Pelotas, 2016.
- BARTH, J.; DYER, C.; STASSINOPOULOS, E. Space, atmospheric, and terrestrial radiation environments. **IEEE Transactions on Nuclear Science**, v. 50, n. 3, p. 466–482, 2003.
- BENSO, A.; CARLO, S. D.; NATALE, G. D.; PRINETTO, P. A watchdog processor to detect data and control flow errors. In: **9th IEEE On-Line Testing Symposium, 2003. IOLTS 2003**. [S. l.: s. n.], 2003. p. 144–148.
- BOTMA, P. J. **The design and development of an ADCS OBC for a CubeSat**. Dissertação (Master of Science in Electrical and Electronic Engineering) – Stellenbosch University, Electrical and Electronic Engineering, Stellenbosch, South Africa, 2011.
- BOTMA, P. J.; BARNARD, A.; STEYN, W. H. Low cost fault tolerant techniques for nano/pico-satellite applications. In: **2013 Africon**. [S. l.: s. n.], 2013. p. 1–5.
- BUSCH, S.; SCHILLING, K. Robust and efficient obdh core module for the flexible picosatellite bus uwe-3. **IFAC Proceedings Volumes**, v. 46, n. 19, p. 218–223, 2013. ISSN 1474-6670. 19th IFAC Symposium on Automatic Control in Aerospace. Disponível em: <https://www.sciencedirect.com/science/article/pii/S1474667015363254>.
- CALDWELL, D. W. **Minimalist fault-tolerance techniques for mitigating single-event effects in non-radiation-hardened microcontrollers**. Tese (Doutorado) – University of California, Los Angeles, jan. 1998.
- CASTRO, H. d. S.; SILVEIRA, J. A. N. da; COELHO, A. A. P.; SILVA, F. G. A. e; MAGALHÃES, P. d. S.; LIMA, O. A. de. A correction code for multiple cells upsets in memory devices for space applications. In: **2016 14th IEEE International New Circuits and Systems Conference (NEWCAS)**. [S. l.: s. n.], 2016. p. 1–4.
- CECCHETTO, M.; ALÍA, R. G.; WROBEL, F.; TALI, M.; STEIN, O.; LERNER, G.; BILKO, K.; ESPOSITO, L.; CASTRO, C. B.; KADI, Y.; DANZECA, S.; BRUCOLI, M.; CAZZANIGA, C.; BAGATIN, M.; GERARDIN, S.; PACCAGNELLA, A. Thermal neutron-induced seus in the lhc accelerator environment. **IEEE Transactions on Nuclear Science**, v. 67, n. 7, p. 1412–1420, 2020.

CORONETTI, A.; ALÌA, R. G.; WANG, J.; TALI, M.; CECCHETTO, M.; CAZZANIGA, C.; JAVANAINEN, A.; SAIGNÉ, F.; LEROUX, P. Assessment of proton direct ionization for the radiation hardness assurance of deep submicron srams used in space applications. **IEEE Transactions on Nuclear Science**, v. 68, n. 5, p. 937–948, 2021.

CUBE SPACE. **Cube Computer V4.1**. 2016. Disponível em: <https://www.cubesatshop.com/wp-content/uploads/2016/06/CubeComputer-V4.1-Datasheet-v1.3.pdf>. Acesso em: 27 out. 2021.

CUI, Y.; ZHANG, X. Research and implementation of interleaving grouping hamming code algorithm. In: **2013 IEEE International Conference on Signal Processing, Communication and Computing (ICSPCC 2013)**. [S. l.: s. n.], 2013. p. 1–4.

DATA PATTERNS. **Data Patterns DP-OBC-0402**. 2021. Disponível em: https://www.datapatternsindia.com/product/onboard_computer.php. Acesso em: 12 nov. 2021.

DSILVA, D.; WANG, J.-J.; REZZAK, N.; JAT, N. Neutron see testing of the 65nm smartfusion2 flash-based fpga. In: **2015 IEEE Radiation Effects Data Workshop (REDW)**. [S. l.: s. n.], 2015. p. 1–5.

DUBROVA, E. **Fault-Tolerant Design**. [S. l.]: Springer Publishing Company, Incorporated, 2013. ISBN 1461421128.

DUGAN, J. B.; BAVUSO, S. J.; BOYD, M. A. Fault trees and markov models for reliability analysis of fault-tolerant digital systems. **Reliability Engineering System Safety**, v. 39, n. 3, p. 291–307, 1993. ISSN 0951-8320. Disponível em: <https://www.sciencedirect.com/science/article/pii/095183209390005J>.

FAYYAZ, M.; VLADIMIROVA, T. Fault-tolerant distributed approach to satellite on-board computer design. In: **2014 IEEE Aerospace Conference**. [S. l.: s. n.], 2014. p. 1–12.

FAYYAZ, M.; VLADIMIROVA, T.; CAUJOLLE, J.-M. Adaptive middleware design for satellite fault-tolerant distributed computing. In: **2012 NASA/ESA Conference on Adaptive Hardware and Systems (AHS)**. [S. l.: s. n.], 2012. p. 23–30.

FREITAS, D.; MOTA, D.; GOERL, R.; MARCON, C.; VARGAS, F.; SILVEIRA, J.; MOTA, J. Pcosa: A product error correction code for use in memory devices targeting space applications. **Integration**, v. 74, p. 71–80, 2020. ISSN 0167-9260. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0167926019305243>.

GAMA, M. A. **Núcleos IP Corretores de Erros Para Proteção de Memória em SoC**. 2008. Dissertação (Mestrado em Engenharia Elétrica) – Programa de Pós-Graduação em Engenharia Elétrica, Pontifícia Universidade Católica do Rio Grande do Sul Faculdade Engenharia, Porto Alegre, 2008.

GAUSS SRL. **Gauss Srl ABACUS**. 2017. Disponível em: <https://www.gaussteam.com/products/onboard-computer/abacus-2/>. Acesso em: 12 nov. 2021.

GEORGE, A. D.; WILSON, C. M. Onboard processing with hybrid and reconfigurable computing on small satellites. **Proceedings of the IEEE**, v. 106, n. 3, p. 458–470, 2018.

GRACIA-MORÁN, J.; SAIZ-ADALID, L. J.; GIL-TOMÁS, D.; GIL-VICENTE, P. J. Improving error correction codes for multiple-cell upsets in space applications. **IEEE Transactions on Very Large Scale Integration (VLSI) Systems**, v. 26, n. 10, p. 2132–2142, 2018.

- HAMMING, R. W. Error detecting and error correcting codes. **The Bell System Technical Journal**, v. 29, n. 2, p. 147–160, 1950.
- HANE, J. S. **A Fault-Tolerant Computer Architecture For Space Vehicle Applications**. 2012. Dissertação (Master of Science in Electrical Engineering) – Montana State University, Electrical Engineering, Bozeman, Montana, 2012.
- ISSI. **IS61WV204816BLL-10TLI Datasheet**. 2016. Disponível em: <https://www.issi.com/WW/pdf/61-64WV204816BLL.pdf>. Acesso em: 16 dez. 2021.
- KUWAHARA, T. **FPGA-based Reconfigurable On-Board Computing Systems For Space Applications**. Tese (Doutorado), 01 2010.
- LAYTON, P.; CZAJKOWSKI, D.; MARSHALL, J.; ANTHONY, H.; BOSS, R. Single event latchup protection of integrated circuits. In: **RADECS 97. Fourth European Conference on Radiation and its Effects on Components and Systems (Cat. No.97TH8294)**. [S. l.: s. n.], 1997. p. 327–331.
- LEPPINEN, H.; KESTILÄ, A.; PIHAJOKI, P.; JOKELAINEN, J.; HAUNIA, T. On-board data handling for ambitious nanosatellite missions using automotive-grade lockstep microcontrollers. In: **Small Satellites Systems and Services - The 4S Symposium 2014**. [S. l.: s. n.], 2014.
- LIM, S. S. L. **A fault tolerant parallel computing architecture for remote sensing satellites**. Tese (Doutorado) – RMIT University, 2009.
- LINEAR TECHNOLOGY. **LT8610EMSEPF Datasheet**. 2021. Disponível em: <https://www.analog.com/media/en/technical-documentation/data-sheets/lt8610.pdf>. Acesso em: 16 dez. 2021.
- MAHMOOD, A.; MCCLUSKEY, E. Concurrent error detection using watchdog processors-a survey. **IEEE Transactions on Computers**, v. 37, n. 2, p. 160–174, 1988.
- MAQBOOL, S. **A system-level supervisory approach to mitigate single event functional interrupts in data handling architectures**. 186 p. Tese (Doutorado) – University of Surrey (United Kingdom), 2006.
- MAXIM INTEGRATED. **MAX8516EUB+T Datasheet**. 2016. Disponível em: <https://datasheets.maximintegrated.com/en/ds/MAX8516-MAX8518.pdf>. Acesso em: 16 dez. 2021.
- MERL, R.; GRAHAM, P. A low-cost, radiation-hardened single-board computer for command and data handling. In: **2016 IEEE Aerospace Conference**. [S. l.: s. n.], 2016. p. 1–8.
- MESSENGER, G. C. Radiation hardening of electronic systems invited paper. **IEEE Transactions on Nuclear Science**, v. 16, n. 6, p. 160–168, 1969.
- MICROSEMI. **M2S010-VFG400I Datasheet**. 2018. Disponível em: https://www.microsemi.com/document-portal/doc_download/132042-ds0128-igloo2-and-smartfusion2-datasheet. Acesso em: 16 dez. 2021.
- MOTA, D. F. M. **OPENOBC: Uma Arquitetura de um Computador de Bordo Open Source e Baixo Custo Para o Padrão CubeSat**. 2017. Dissertação (Mestrado em Engenharia de Teleinformática) – Centro de Tecnologia, Programa de Pós-Graduação em Engenharia de Teleinformática, Universidade Federal do Ceará, Fortaleza, 2017.

NAMJOO, M.; MCCLUSKEY, E. Watchdog processors and capability checking. In: **Twenty-Fifth International Symposium on Fault-Tolerant Computing, 1995, 'Highlights from Twenty-Five Years'**. [S. l.: s. n.], 1995. p. 94–.

NICOLAIDIS, M. A low-cost single-event latchup mitigation scheme. In: **12th IEEE International On-Line Testing Symposium (IOLTS'06)**. [S. l.: s. n.], 2006. p. 5 pp.–.

NICOLAIDIS, M.; VARGAS, F.; COURTOIS, B. Design of built-in current sensors for concurrent checking in radiation environments. **IEEE Transactions on Nuclear Science**, v. 40, n. 6, p. 1584–1590, 1993.

NOELDEKE, C.; BOETTCHER, M.; MOHR, U.; GAISSER, S.; Alvarez Rúa, M.; EICKHOFF, J.; LESLIE, M.; Von Thun, M.; KLINKNER, S.; VARATHARAJOO, R. Single event upset investigations on the “flying laptop” satellite mission. **Advances in Space Research**, v. 67, n. 6, p. 2000–2009, 2021. ISSN 0273-1177. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0273117720309054>.

OHLSSON, J.; RIMEN, M.; GUNNEFLO, U. A study of the effects of transient fault injection into a 32-bit risc with built-in watchdog. In: **[1992] Digest of Papers. FTCS-22: The Twenty-Second International Symposium on Fault-Tolerant Computing**. [S. l.: s. n.], 1992. p. 316–325.

POGHOSYAN, A.; GOLKAR, A. Cubesat evolution: Analyzing cubesat capabilities for conducting science missions. **Progress in Aerospace Sciences**, v. 88, p. 59–83, 2017. ISSN 0376-0421. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0376042116300951>.

PRADHAN, D. K. (Ed.). **Fault-Tolerant Computer System Design**. USA: Prentice-Hall, Inc., 1996. ISBN 0130578878.

RADAELLI, D.; PUCHNER, H.; WONG, S.; DANIEL, S. Investigation of multi-bit upsets in a 150 nm technology sram device. **IEEE Transactions on Nuclear Science**, v. 52, n. 6, p. 2433–2437, 2005.

RAZZAGHI, E. **Design and Qualification of On-Board Computer for Aalto-1 CubeSat**. 66 p. Dissertação (Mestrado), 2012. Validerat; 20120828 (anonymous).

REZZAK, N.; WANG, J.-J.; HUANG, C.-K.; NGUYEN, V.; BAKKER, G. Total ionizing dose characterization of 65 nm flash-based fpga. In: **2014 IEEE Radiation Effects Data Workshop (REDW)**. [S. l.: s. n.], 2014. p. 1–5.

SATOH, S.; TOSAKA, Y.; WENDER, S. Geometric effect of multiple-bit soft errors induced by cosmic ray neutrons on dram's. **IEEE Electron Device Letters**, v. 21, n. 6, p. 310–312, 2000.

SECONDO, R.; ALIA, R. G.; PERONNARD, P.; BRUGGER, M.; MASI, A.; DANZECA, S.; FERRARO, R.; MERLENGHI, A.; VAILLE, J. R.; DUSSEAU, L. Analysis of sel on commercial sram memories for latchup detection and protection in leo space applications. In: **2016 16th European Conference on Radiation and Its Effects on Components and Systems (RADECS)**. [S. l.: s. n.], 2016. p. 1–5.

SHOGA, M.; BINDER, D. Theory of single event latchup in complementary metal-oxide semiconductor integrated circuits. **IEEE Transactions on Nuclear Science**, v. 33, n. 6, p. 1714–1717, 1986.

SILVA, F. G. A. **Um Código Extensível Para Correção de Multiple Bit Upsets em Memórias**. 2018. Dissertação (Mestrado em Engenharia de Teleinformática) – Centro de Tecnologia, Programa de Pós-Graduação em Engenharia de Teleinformática, Universidade Federal do Ceará, Fortaleza, 2018.

SORIN, D. J. **Fault Tolerant Computer Architecture**. [S. l.]: Morgan and Claypool Publishers, 2009. ISBN 1598299530.

TEXAS INSTRUMENTS. **TMS5701114CZWTQQ1 Datasheet**. 2015. Disponível em: <https://www.ti.com/lit/gpn/TMS570LS1114>. Acesso em: 16 dez. 2021.

THE CUBESAT STANDARD. **The CubeSat Standard**. 2021. Disponível em: <https://www.cubesat.org/about>. Acesso em: 11 out. 2021.

TIAN, S.; YIN, Z.; YAN, J.; LIU, X. Design and implementation of a low-cost fault-tolerant on-board computer for micro-satellite. In: **7th International Conference on Communications and Networking in China**. [S. l.: s. n.], 2012. p. 129–134.

VARGAS, F. **Improving Electronics Reliability for Space Systems Based on Current Monitoring**. Tese (Doutorado) – Laboratoire TIMA, Institut National Polytechnique de Grenoble (INPG), França, 1995.

VARGAS, F.; NICOLAIDIS, M. Seu-tolerant sram design based on current monitoring. In: **Proceedings of IEEE 24th International Symposium on Fault-Tolerant Computing**. [S. l.: s. n.], 1994. p. 106–115.

VINCI, E.; SAOTOME, O. Reliability of on-board computer for itasat university satellite. In: **2010 11th Latin American Test Workshop**. [S. l.: s. n.], 2010. p. 1–3.

WERTZ, J.; LARSON, W. **Space Mission Analysis and Design**. Springer Netherlands, 1999. (Space Technology Library). ISBN 9780792359012. Disponível em: <https://books.google.com.br/books?id=veyGEAKFbiYC>.

YANG, M.; HUA, G.; FENG, Y.; GONG, J. **Fault-Tolerance Techniques for Spacecraft Control Computers**. 1st. ed. [S. l.]: Wiley Publishing, 2017. ISBN 111910727X.

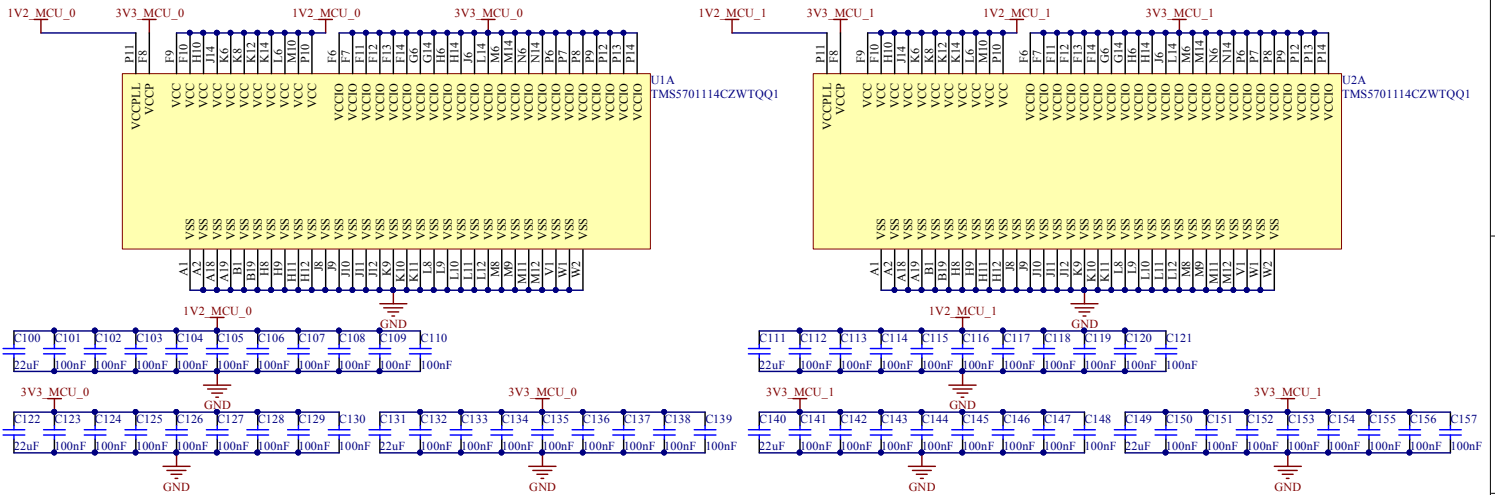
YASHIRO, H.; FUJIWARA, T.; MORI, K. A high assurance on-line recovery technology for a space on-board computer. In: **Proceedings 5th International Symposium on Autonomous Decentralized Systems**. [S. l.: s. n.], 2001. p. 47–56.

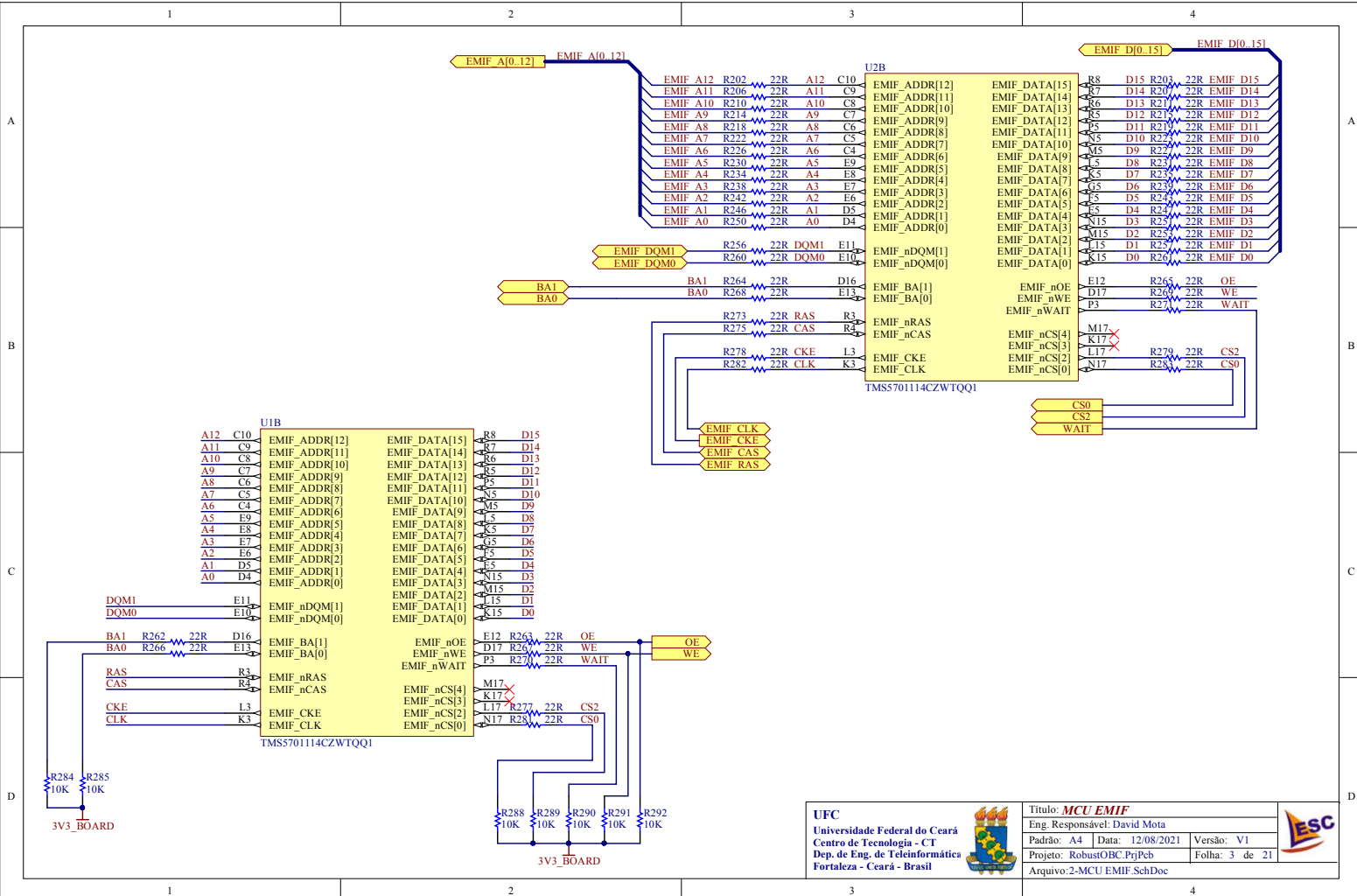
YUEN, B.; SIMA, M. Low cost radiation hardened software and hardware implementation for cubesats. **The Arbutus Review**, v. 9, p. 46–62, 09 2018.

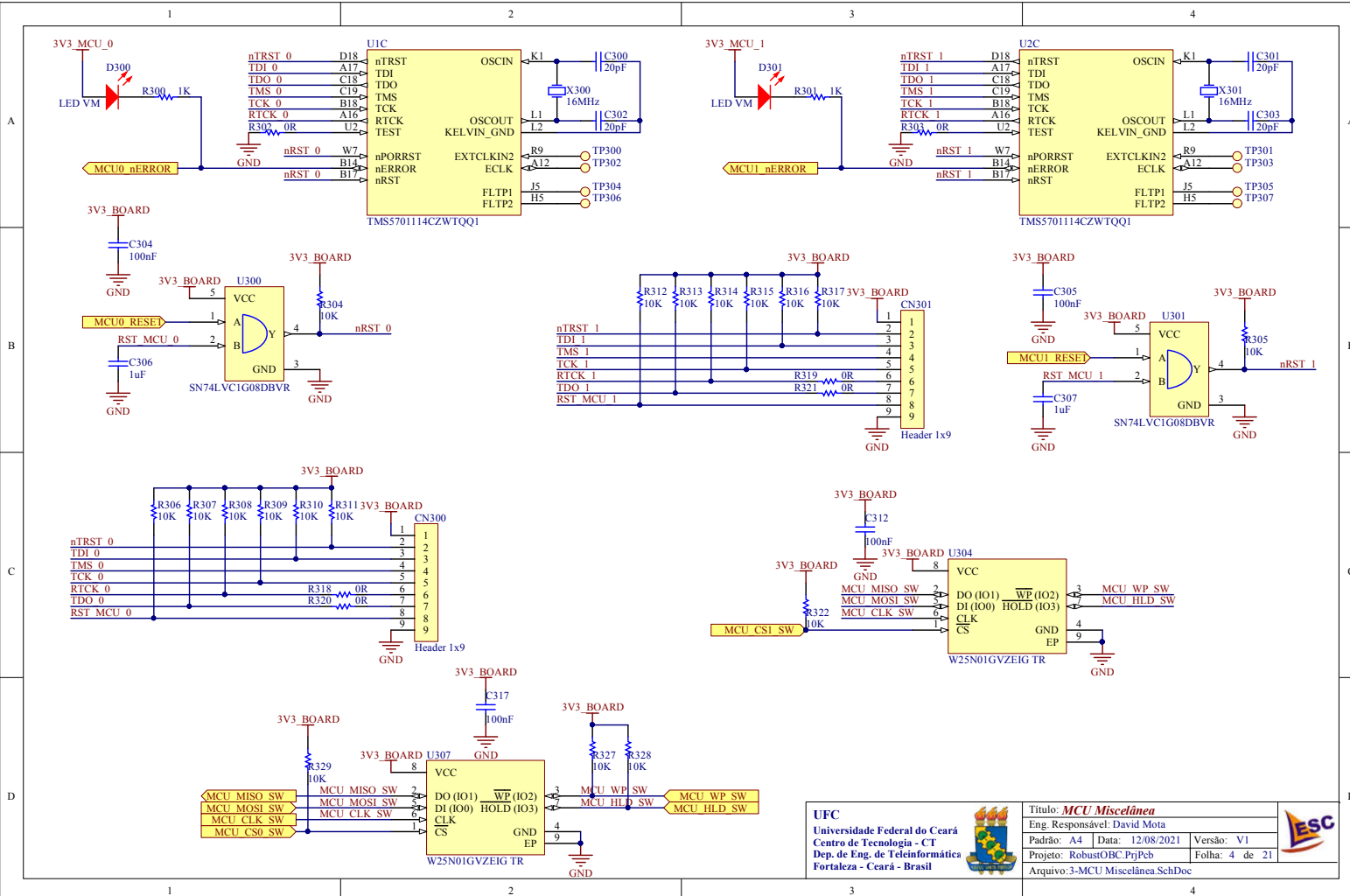
ZIN TECHNOLOGIES. **ZIN Space Qualified Single Board Computer SBCFA1000**. 2021. Disponível em: https://0d088610-0fd0-4d53-8a7d-560afcaae937.filesusr.com/ugd/6ed110_1958097a293241c58d3abf883eb0aa98.pdf. Acesso em: 20 out. 2021.

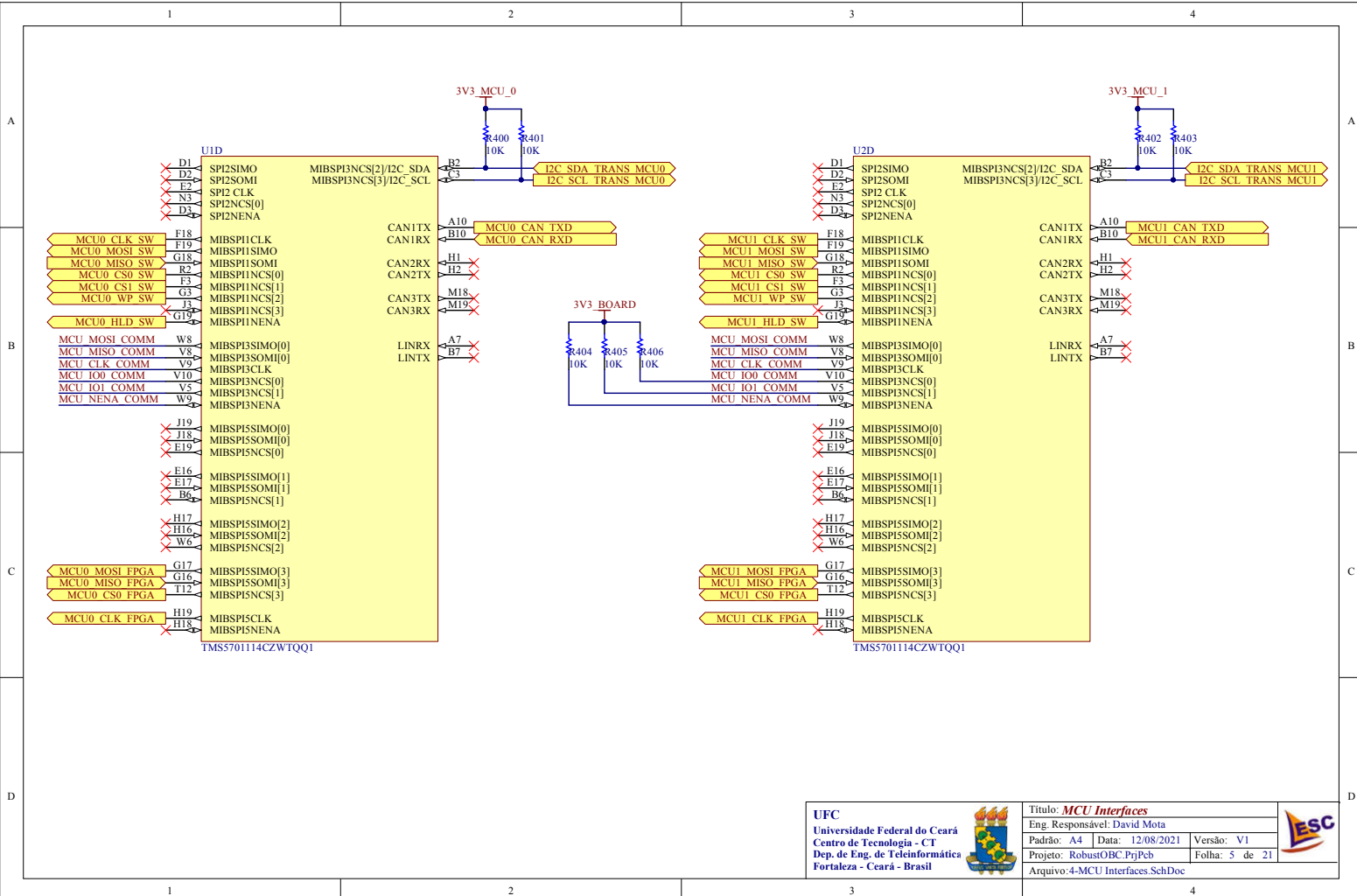
APÊNDICE A – ESQUEMAS ELÉTRICOS

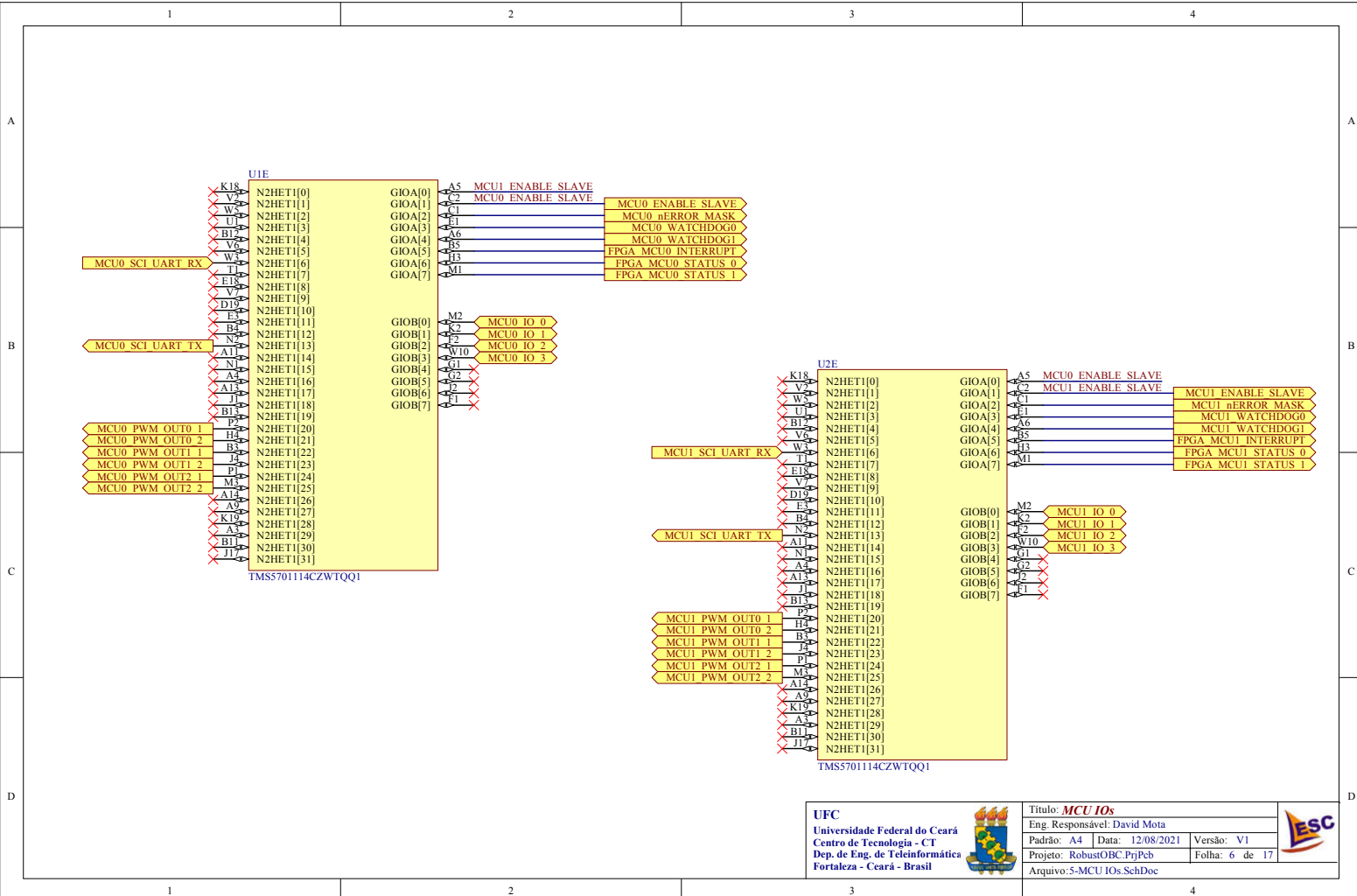
Esquemas elétricos desenvolvidos para a implementação da placa de circuito impressa do Robust OBC.

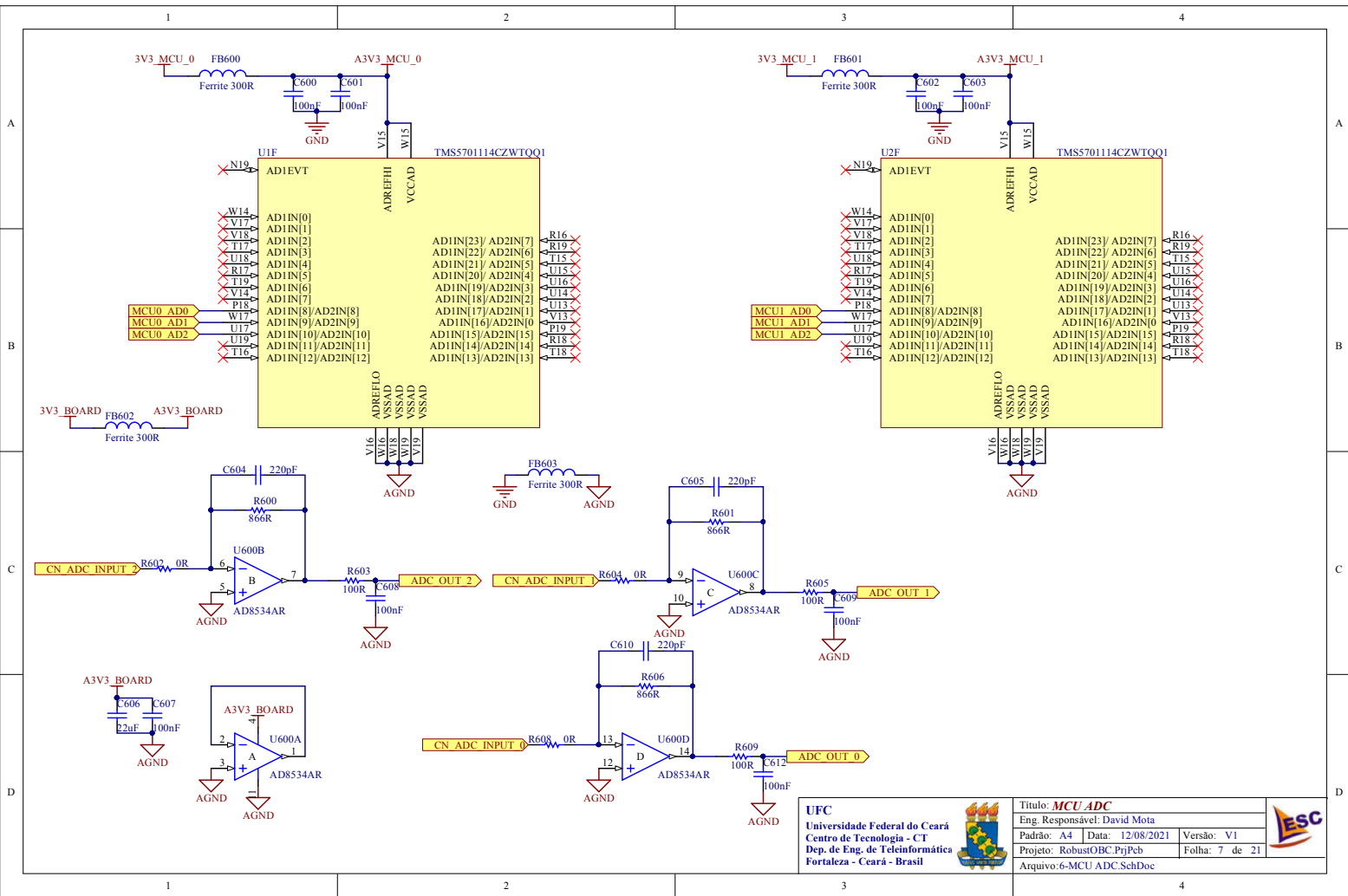










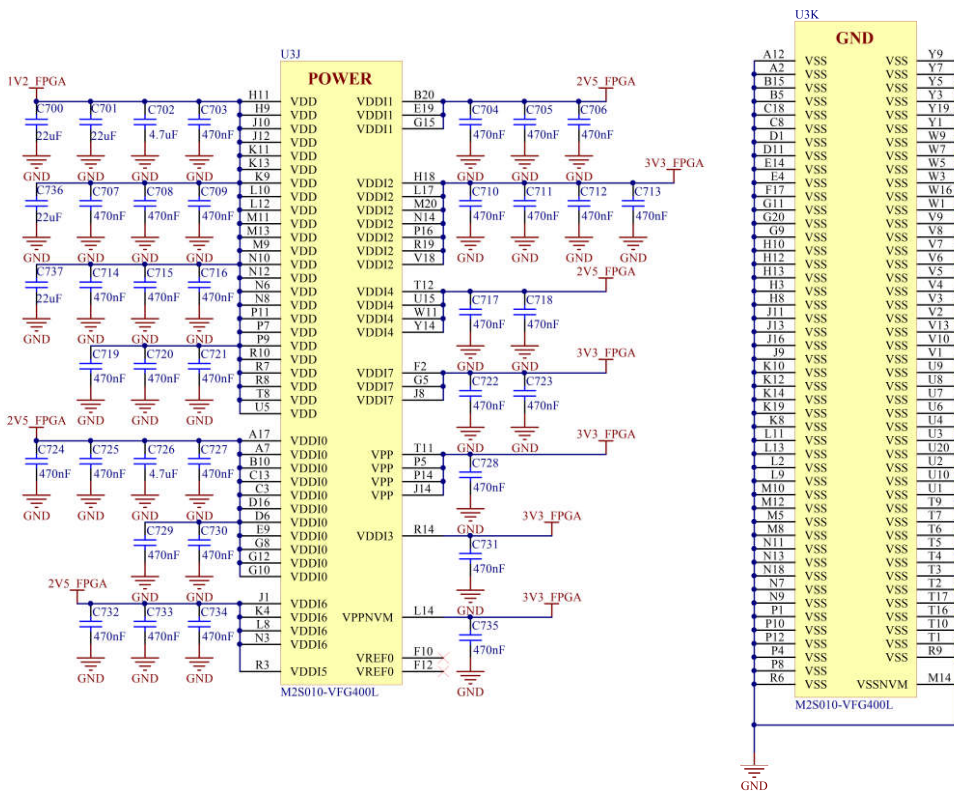


UFC
 Universidade Federal do Ceará
 Centro de Tecnologia - CT
 Dep. de Eng. de Teleinformática
 Fortaleza - Ceará - Brasil



Título: MCU ADC	
Eng. Responsável: David Mota	
Padrão: A4	Data: 12/08/2021
Projeto: RobustOBC.PrjPcb	Folha: 7 de 21
Arquivo: 6-MCU ADC.SchDoc	

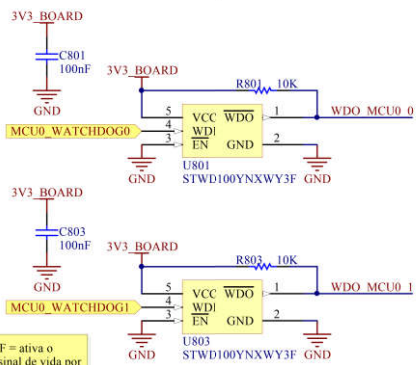
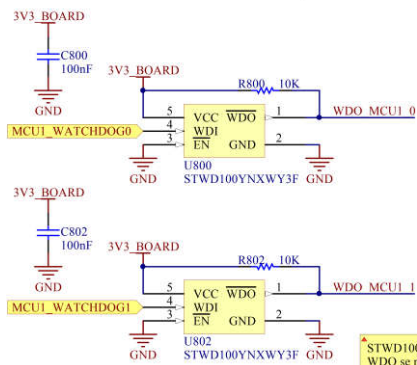




Doc AC396:

2.1 Power Supply Sequencing and Power-On Reset
 All the I/Os are held in a high-impedance state by the system controller until all power supplies are at their required levels and the system controller has completed the reset sequence. The power-on reset circuitry in SmartFusion2/IGLOO2 devices require the VDD and VPP supplies to ramp monotonically from 0 V to the minimum recommended operating voltage within a predefined time. There is no sequencing requirement on VDD and VPP. Four ramp rate options are available during design generation: 50 μ s, 1 ms, 10 ms, and 100 ms. Each selection represents the maximum ramp rate to apply to VDD and VPP.

2.2 I/O State During Power-Up and Power-Down
I/O State During Power-Up and Power-Down
 Before powering up, all SmartFusion2/IGLOO2 I/Os are in tri-state mode. These I/Os remain in tri-state mode during power-up until the voltage supplies (VDD and VDDI) have reached their functional levels. After VDD and VDDI reach the functional level, the outputs exit the tri-state mode and are driven to the value determined by the design. The behavior of SmartFusion2/IGLOO2 I/Os is independent of the VDD and VDDI sequence. Figure 1, page 3 shows a scenario where VDD is powered first, and followed by VDDI. Figure 2, page 3 shows the scenario where VDDI is powered first, and followed by VDD.



STWD100YNXWY3F = ativa o WDO se não receber sinal de vida por pelo menos 71ms

SAIDA TWU:
 SUPPLY ENABLE = ATIVO ALTO
 RESET = ATIVO BAIXO

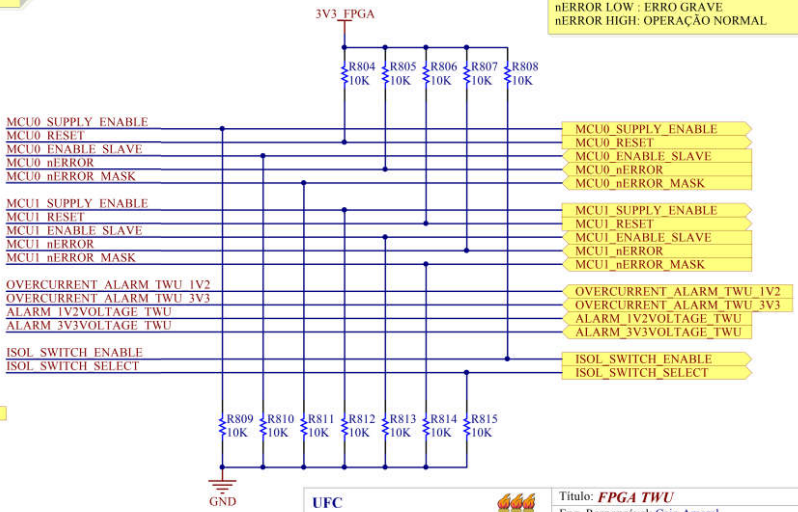
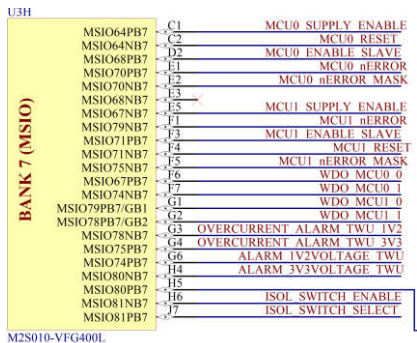
ENTRADA TWU:
 ENABLE SLAVE = ATIVO ALTO;
 MCU_NERROR = ATIVO BAIXO
 MCU_NERROR_MASK = ATIVO ALTO

WDO = ATIVO BAIXO
 OVERCURRENT = ATIVO BAIXO
 VOLTAGE_ALARM = ATIVO ALTO

ISOL_SWITCH_ENABLE = ATIVO BAIXO
 ISOL_SWITCH_SELECT = LOW:NC, HIGH:NO

nERROR e nERROR_MASK: OR de ambos?
 nERROR_MASK LOW : OPERAÇÃO NORMAL
 nERROR_MASK HIGH: MASCARANDO SINAL nERROR

nERROR LOW: ERRO GRAVE
 nERROR HIGH: OPERAÇÃO NORMAL

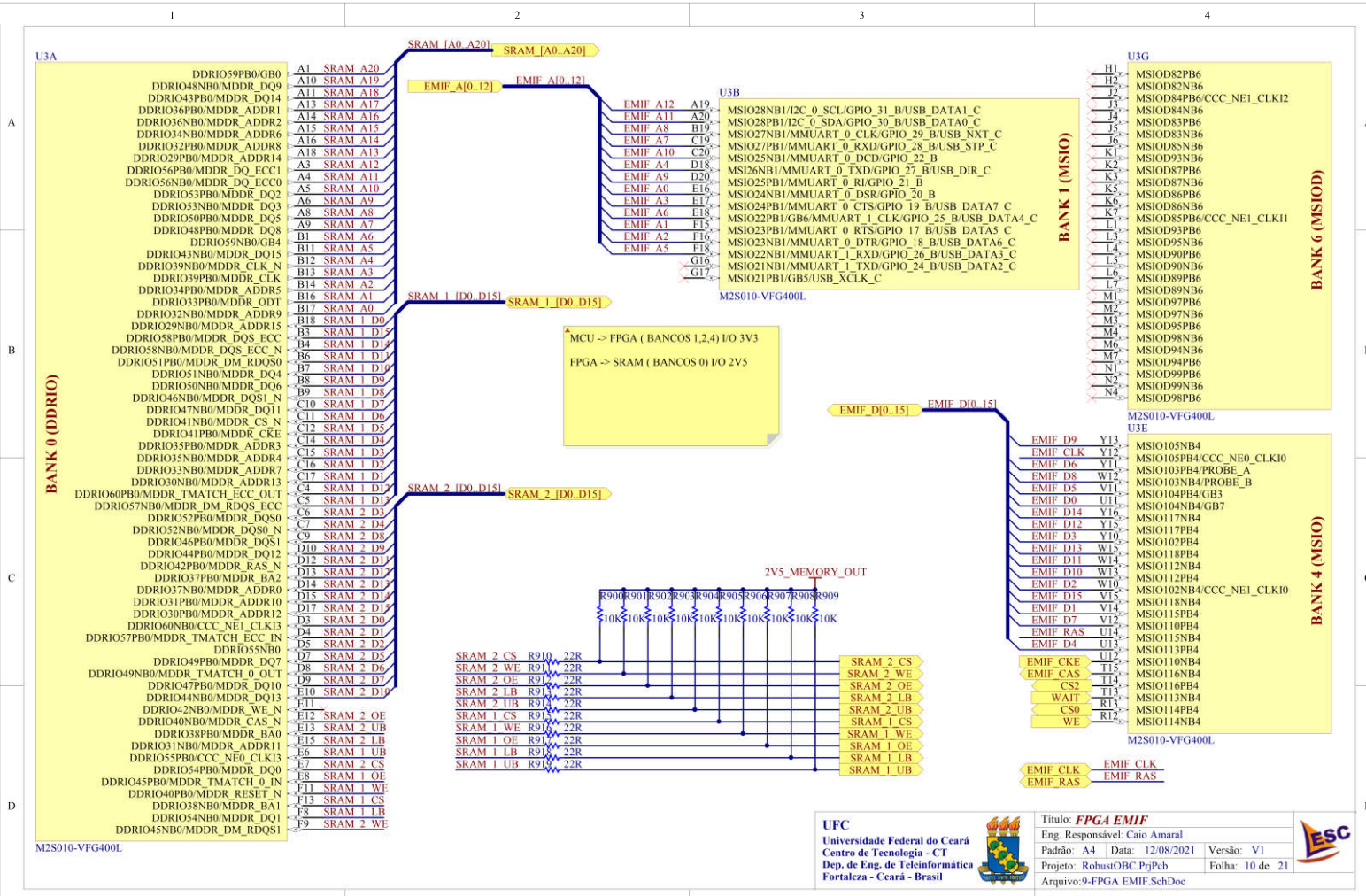


UFC
 Universidade Federal do Ceará
 Centro de Tecnologia - CT
 Dep. de Eng. de Teleinformática
 Fortaleza - Ceará - Brasil



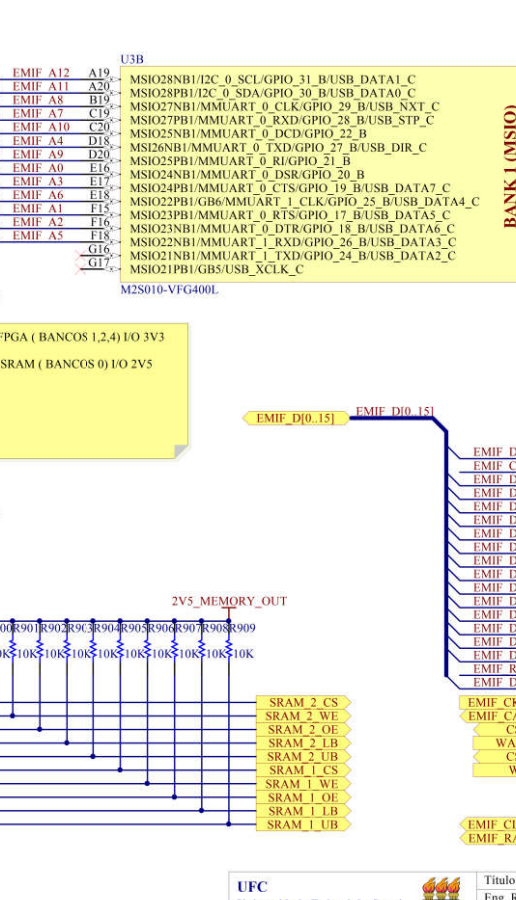
Título: **FPGA TWU**
 Eng. Responsável: Caio Amaral
 Padrão: A4 | Data: 12/08/2021 | Versão: V1
 Projeto: RobustOBC.PrjPeb | Folha: 9 de 21
 Arquivo: 8-FPGA TWU.SchDoc





- BANK 0 (DDRIO)**
- DDRIO59PB0/GB0
 - DDRIO48NB0/MDDR_D09
 - DDRIO43PB0/MDDR_DQ14
 - DDRIO36PB0/MDDR_ADDR1
 - DDRIO36NB0/MDDR_ADDR2
 - DDRIO34NB0/MDDR_ADDR6
 - DDRIO32PB0/MDDR_ADDR8
 - DDRIO29PB0/MDDR_ADDR14
 - DDRIO56PB0/MDDR_DQ_ECC1
 - DDRIO56NB0/MDDR_DQ_ECC0
 - DDRIO53PB0/MDDR_DQ2
 - DDRIO53NB0/MDDR_DQ3
 - DDRIO50PB0/MDDR_DQ5
 - DDRIO48PB0/MDDR_DQ8
 - DDRIO59NB0/GB4
 - DDRIO43NB0/MDDR_DQ15
 - DDRIO39NB0/MDDR_CLK_N
 - DDRIO39PB0/MDDR_CLK
 - DDRIO34PB0/MDDR_ADDR5
 - DDRIO33PB0/MDDR_O01
 - DDRIO32NB0/MDDR_ADDR9
 - DDRIO29NB0/MDDR_ADDR15
 - DDRIO58PB0/MDDR_DQS_ECC
 - DDRIO58NB0/MDDR_DQS_ECC_N
 - DDRIO51PB0/MDDR_DM_RDQ50
 - DDRIO51NB0/MDDR_DQ4
 - DDRIO50NB0/MDDR_DQ6
 - DDRIO46NB0/MDDR_DQ51
 - DDRIO47NB0/MDDR_DQ11
 - DDRIO41NB0/MDDR_CS_N
 - DDRIO41PB0/MDDR_CKE
 - DDRIO35PB0/MDDR_ADDR3
 - DDRIO35NB0/MDDR_ADDR4
 - DDRIO33NB0/MDDR_ADDR7
 - DDRIO30NB0/MDDR_ADDR13
 - DDRIO60PB0/MDDR_TMATCH_ECC_OUT
 - DDRIO57NB0/MDDR_DM_RDQ5_ECC
 - DDRIO52PB0/MDDR_DQ50
 - DDRIO52NB0/MDDR_DQ50_N
 - DDRIO46PB0/MDDR_DQ51
 - DDRIO44PB0/MDDR_DQ12
 - DDRIO42PB0/MDDR_RAS_N
 - DDRIO37PB0/MDDR_BA2
 - DDRIO37NB0/MDDR_ADDR0
 - DDRIO31PB0/MDDR_ADDR10
 - DDRIO30PB0/MDDR_ADDR12
 - DDRIO60NB0/CCC_NE1_CLK13
 - DDRIO57PB0/MDDR_TMATCH_ECC_IN
 - DDRIO55NB0
 - DDRIO49PB0/MDDR_DQ7
 - DDRIO49NB0/MDDR_TMATCH_0_OUT
 - DDRIO47PB0/MDDR_DQ10
 - DDRIO44NB0/MDDR_DQ13
 - DDRIO42NB0/MDDR_WE_N
 - DDRIO40NB0/MDDR_CAS_N
 - DDRIO38PB0/MDDR_BA0
 - DDRIO31NB0/MDDR_ADDR11
 - DDRIO55PB0/CCC_NE0_CLK13
 - DDRIO54PB0/MDDR_DQ0
 - DDRIO45PB0/MDDR_TMATCH_0_IN
 - DDRIO40PB0/MDDR_RESET_N
 - DDRIO38NB0/MDDR_BA1
 - DDRIO54NB0/MDDR_DQ1
 - DDRIO45NB0/MDDR_DM_RDQ51

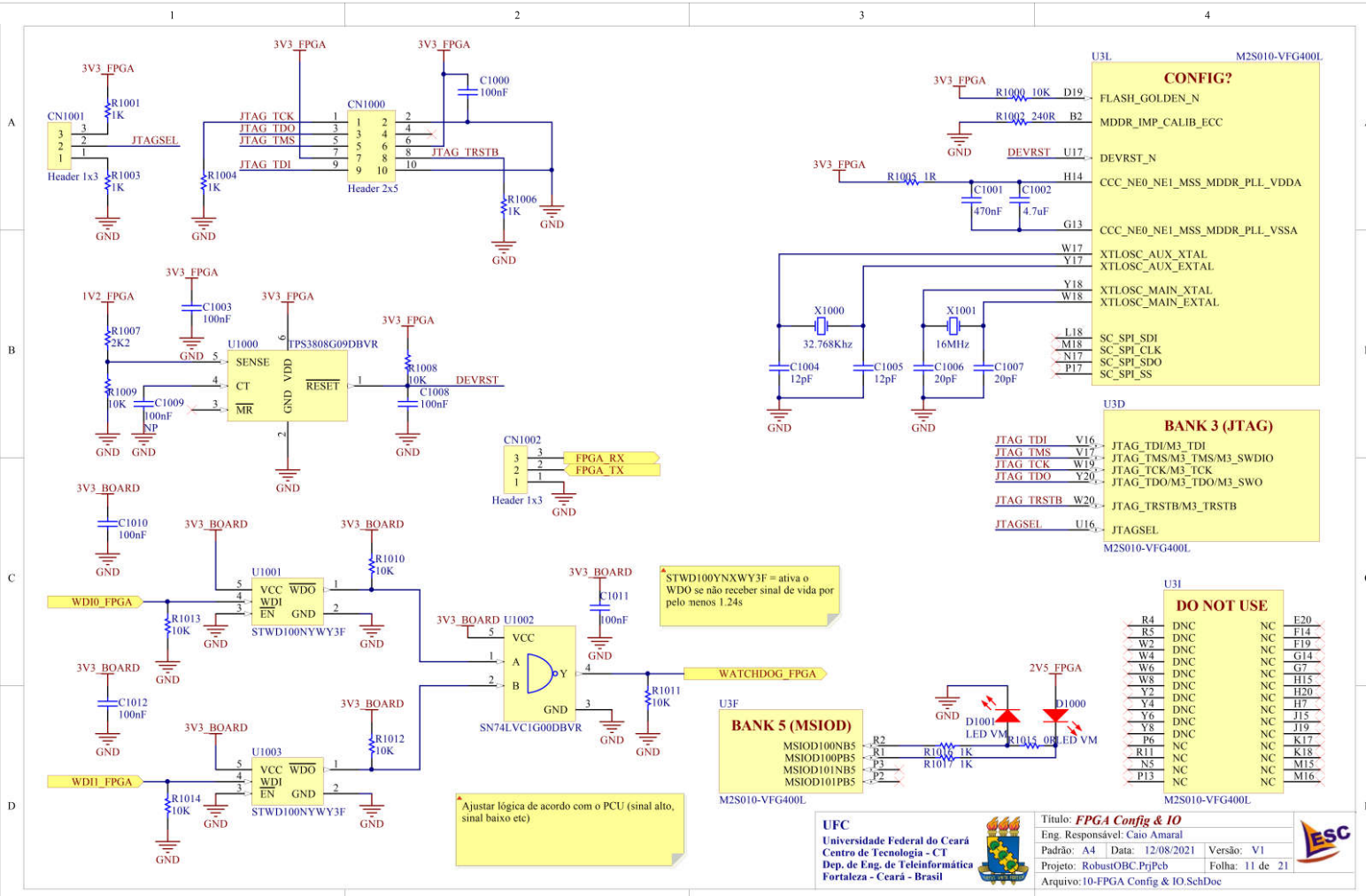
- SRAM [A0..A20]**
- A1 SRAM A20
 - A10 SRAM A19
 - A11 SRAM A18
 - A13 SRAM A17
 - A14 SRAM A16
 - A15 SRAM A15
 - A16 SRAM A14
 - A18 SRAM A13
 - A3 SRAM A12
 - A4 SRAM A11
 - A5 SRAM A10
 - A6 SRAM A9
 - A8 SRAM A8
 - A9 SRAM A7
 - B1 SRAM A6
 - B12 SRAM A4
 - B13 SRAM A3
 - B14 SRAM A2
 - B16 SRAM A1
 - B17 SRAM A0
 - B18 SRAM 1 D0
 - B3 SRAM 1 D1
 - B4 SRAM 1 D1
 - B6 SRAM 1 D1
 - B7 SRAM 1 D1
 - B8 SRAM 1 D9
 - B9 SRAM 1 D8
 - C10 SRAM 1 D7
 - C11 SRAM 1 D6
 - C12 SRAM 1 D5
 - C14 SRAM 1 D4
 - C15 SRAM 1 D3
 - C16 SRAM 1 D2
 - C17 SRAM 1 D1
 - C4 SRAM 1 D1
 - C5 SRAM 1 D1
 - C6 SRAM 2 D3
 - C7 SRAM 2 D4
 - C9 SRAM 2 D8
 - D10 SRAM 2 D9
 - D12 SRAM 2 D1
 - D13 SRAM 2 D1
 - D14 SRAM 2 D1
 - D15 SRAM 2 D1
 - D17 SRAM 2 D1
 - D3 SRAM 2 D0
 - D4 SRAM 2 D1
 - D5 SRAM 2 D2
 - D7 SRAM 2 D5
 - D8 SRAM 2 D6
 - D9 SRAM 2 D7
 - E10 SRAM 2 D10
 - E11
 - E12 SRAM 2 OE
 - E13 SRAM 2 UB
 - E15 SRAM 2 LB
 - E6 SRAM 1 UB
 - E7 SRAM 2 CS
 - E8 SRAM 1 OE
 - F11 SRAM 1 WE
 - F13 SRAM 1 CS
 - F8 SRAM 1 LB
 - F9 SRAM 2 WE

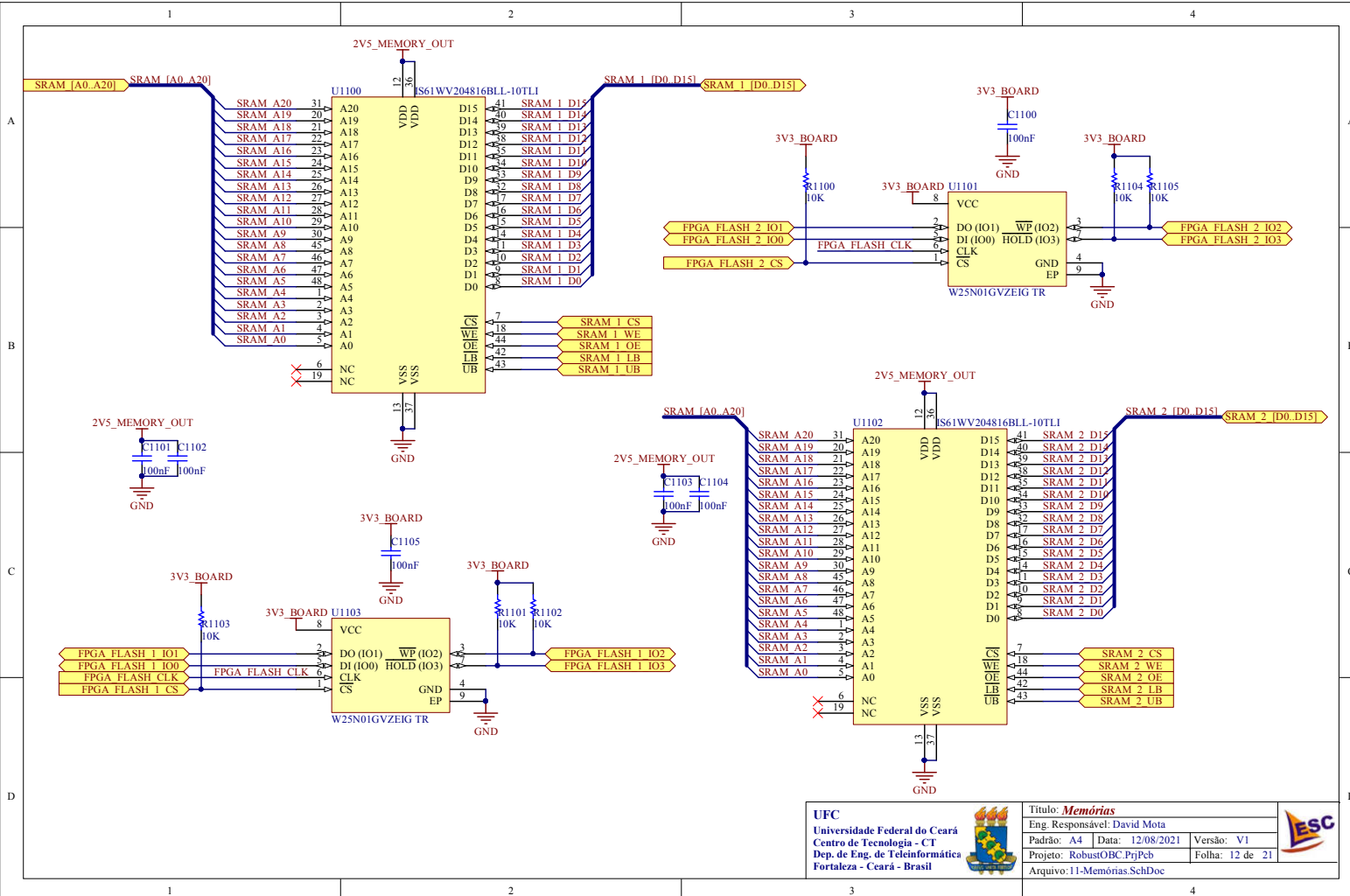


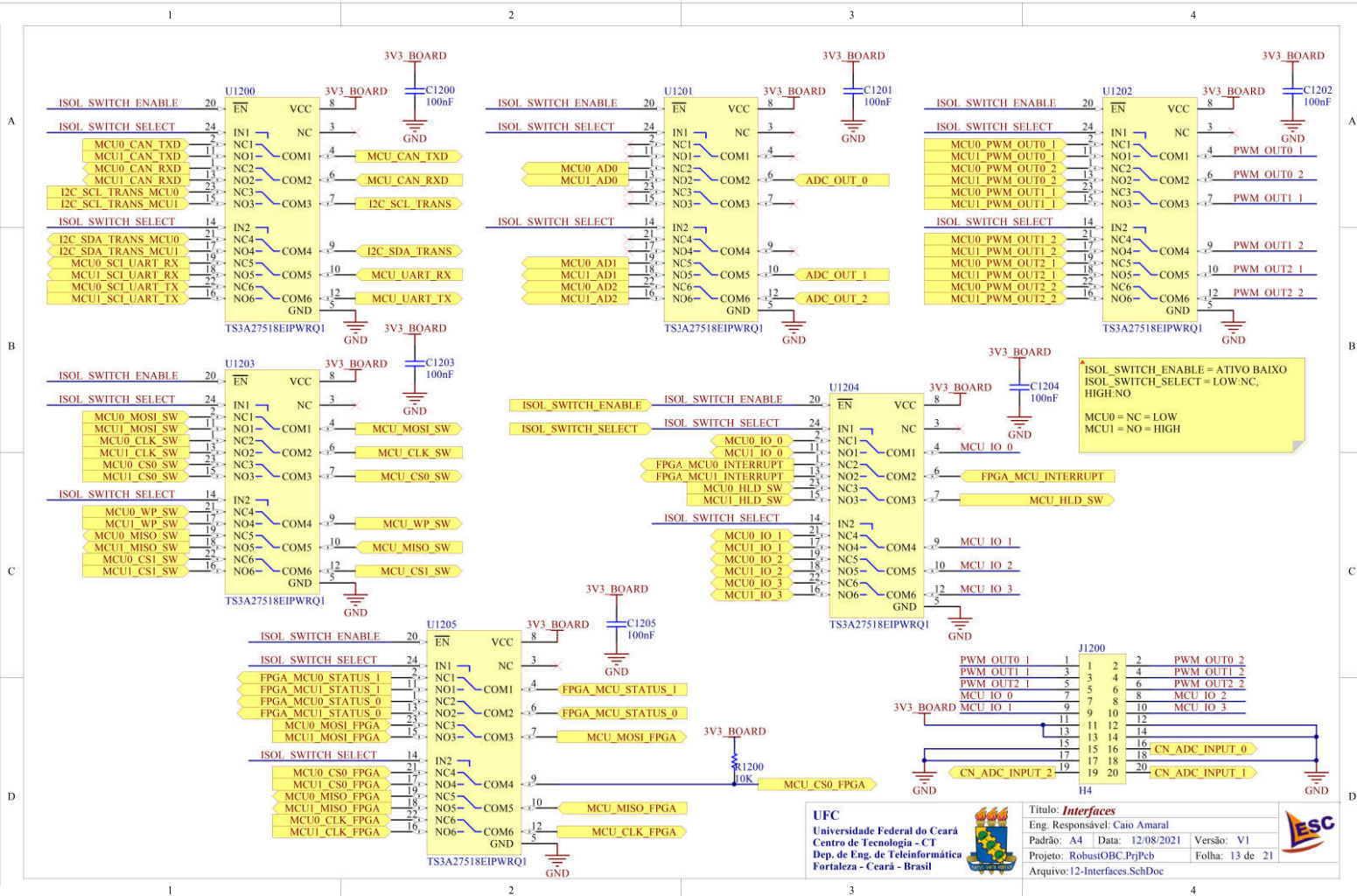
UFC
 Universidade Federal do Ceará
 Centro de Tecnologia - CT
 Dep. de Eng. de Telecomunicações
 Fortaleza - Ceará - Brasil

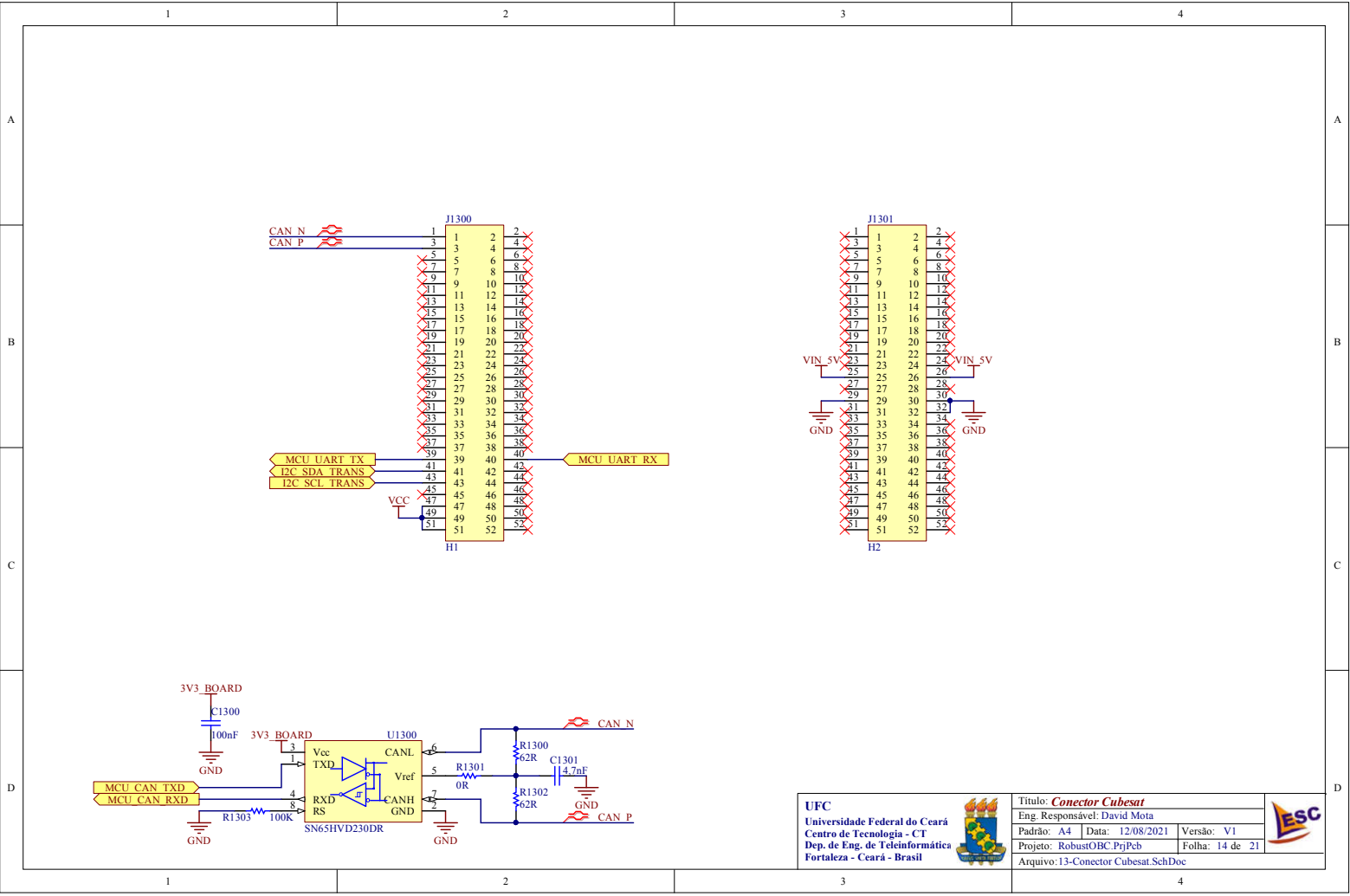
ES

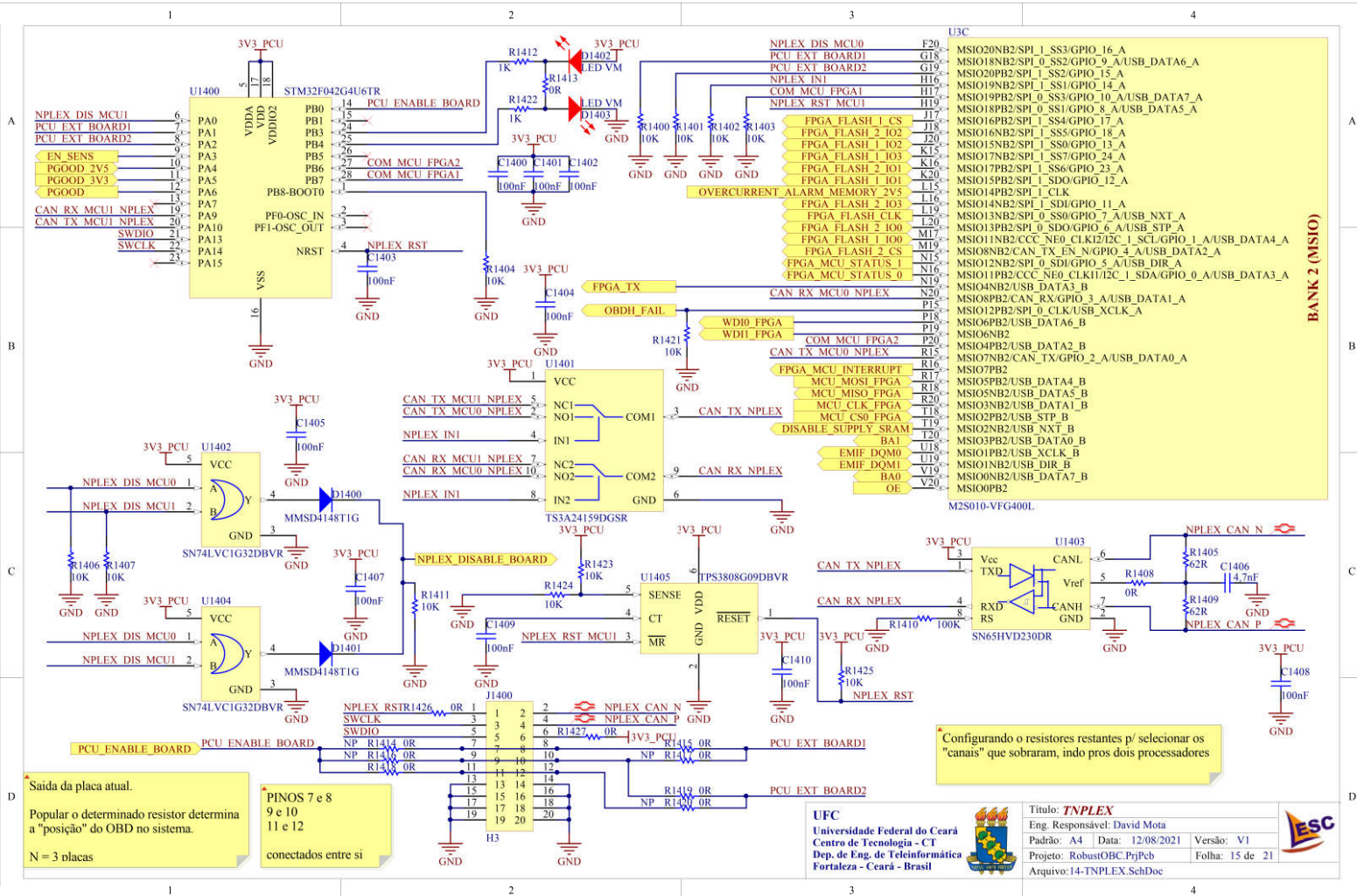
Título: FPGA EMIF
 Eng. Responsável: Caio Amaral
 Padrão: A4 Data: 12/08/2021 Versão: V1
 Projeto: RobustOBC.PrijPeb Folha: 10 de 21
 Arquivo:9-FPGA EMIF.SchDoc







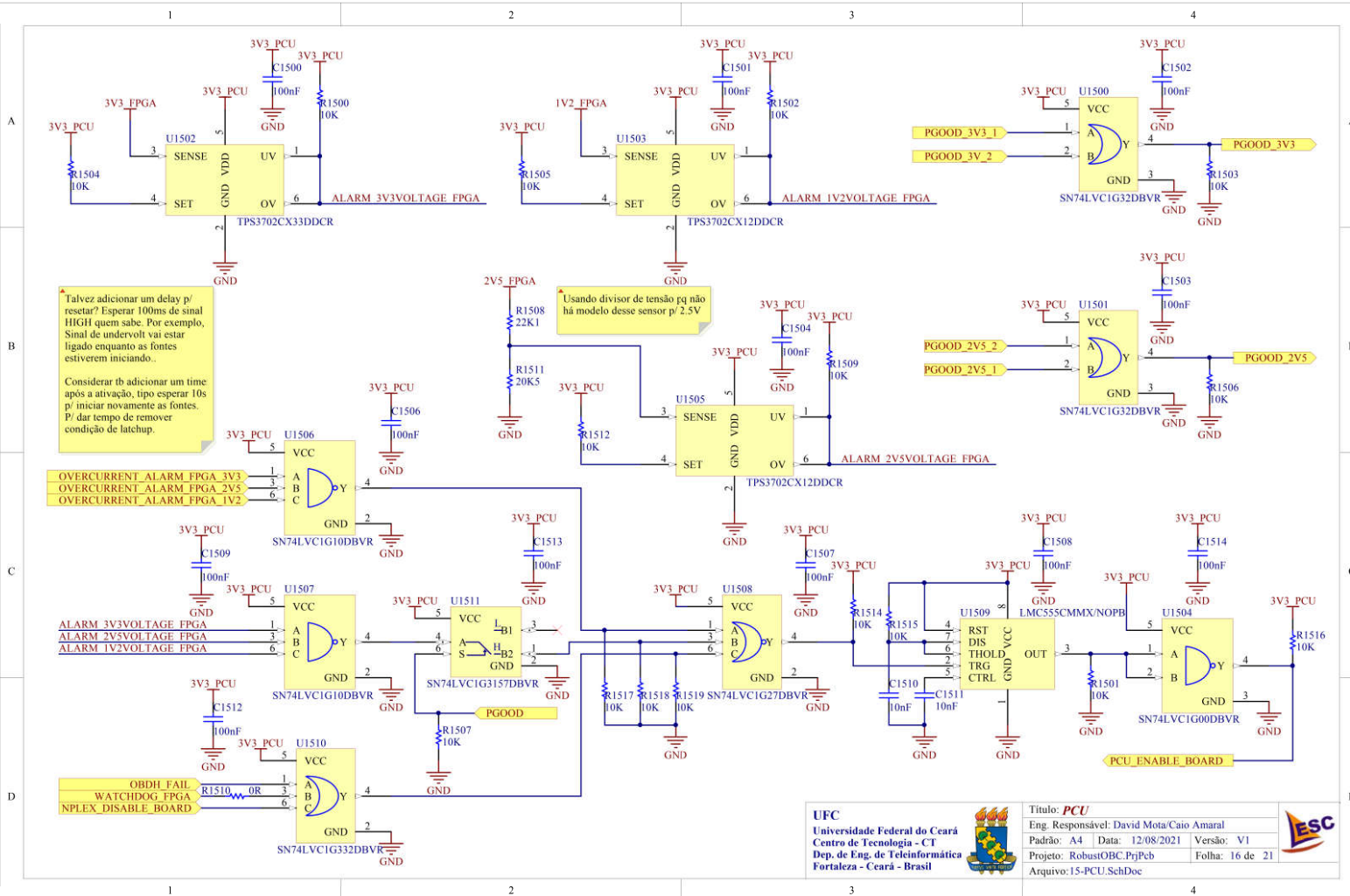




Saída da placa atual.
Popular o determinado resistor determina a "posição" do OBD no sistema.
N = 3 placas

PINOS 7 e 8
9 e 10
11 e 12
conectados entre si

Configurando o resistores restantes p/ selecionar os "canais" que sobram, indo pros dois processadores



Talvez adicionar um delay p/ resetar? Esperar 100ms de sinal HIGH quem sabe. Por exemplo, Sinal de undervolt vai estar ligado enquanto as fontes estiverem iniciando.

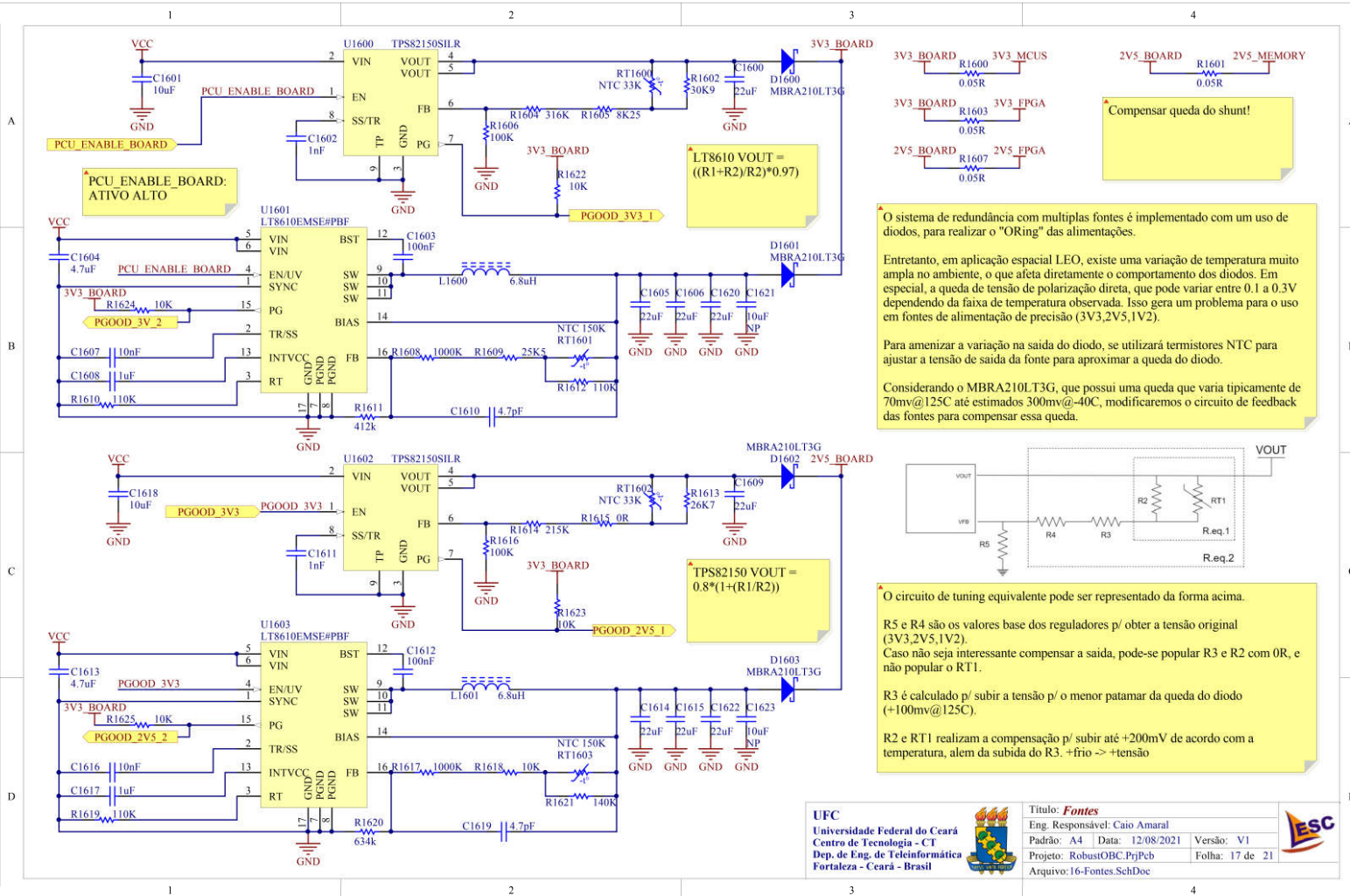
Considerar tb adicionar um time após a ativação, tipo esperar 10s p/ iniciar novamente as fontes. P/ dar tempo de remover condição de latchup.

OVERCURRENT_ALARM_FPGA_3V3
 OVERCURRENT_ALARM_FPGA_2V5
 OVERCURRENT_ALARM_FPGA_1V2

ALARM_3V3VOLTAGE_FPGA
 ALARM_2V5VOLTAGE_FPGA
 ALARM_1V2VOLTAGE_FPGA

OBDH_FAIL_WATCHDOG_FPGA
 NPLEX_DISABLE_BOARD

Usando divisor de tensão pq não há modelo desse sensor p/ 2.5V

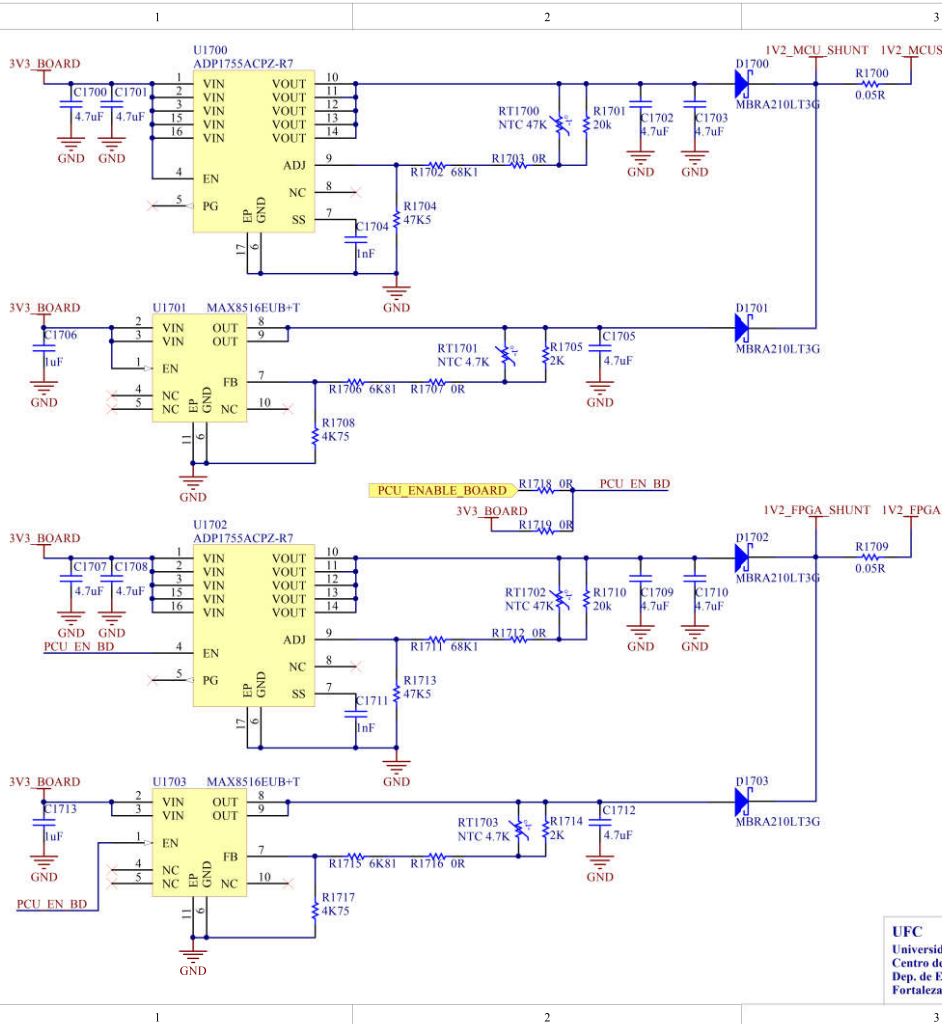


UFC
 Universidade Federal do Ceará
 Centro de Tecnologia - CT
 Dep. de Eng. de Teleinformática
 Fortaleza - Ceará - Brasil



Título: Fontes
 Eng. Responsável: Caio Amaral
 Padrão: A4 Data: 12/08/2021 Versão: V1
 Projeto: RobustOBC.PrjPeb Folha: 17 de 21
 Arquivo:16-Fontes.SchDoc



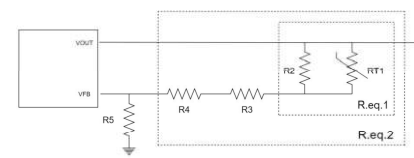


▲ O sistema de redundância com múltiplas fontes é implementado com um uso de diodos, para realizar o "ORing" das alimentações.

Entretanto, em aplicação espacial LEO, existe uma variação de temperatura muito ampla no ambiente, o que afeta diretamente o comportamento dos diodos. Em especial, a queda de tensão de polarização direta, que pode variar entre 0.1 a 0.3V dependendo da faixa de temperatura observada. Isso gera um problema para o uso em fontes de alimentação de precisão (3V3,2V5,1V2).

Para amenizar a variação na saída do diodo, se utilizará termistores NTC para ajustar a tensão de saída da fonte para aproximar a queda do diodo.

Considerando o MBRA210LT3G, que possui uma queda que varia tipicamente de 70mv@125C até estimados 300mv@-40C, modificaremos o circuito de feedback das fontes para compensar essa queda.



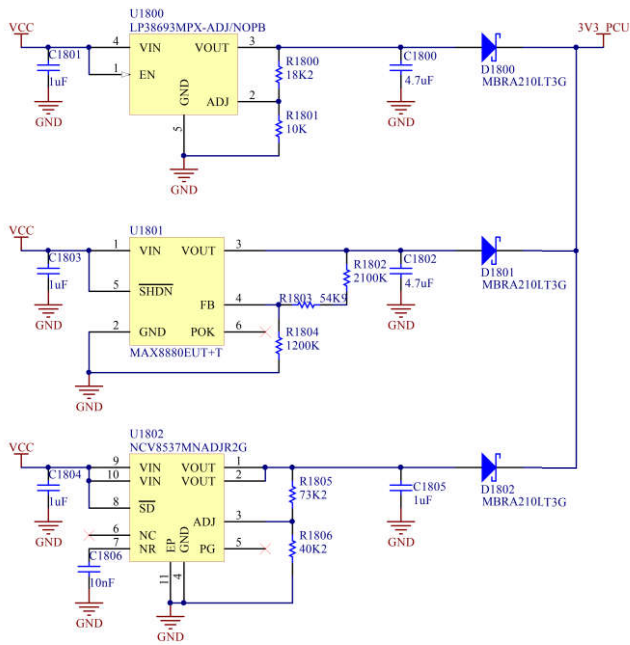
▲ O circuito de tuning equivalente pode ser representado da forma acima.

R5 e R4 são os valores base dos reguladores p/ obter a tensão original (3V3,2V5,1V2).
 Caso não seja interessante compensar a saída, pode-se popular R3 e R2 com 0R, e não popular o RT1.

R3 é calculado p/ subir a tensão p/ o menor patamar da queda do diodo (+100mv@125C).

R2 e RT1 realizam a compensação p/ subir até +200mV de acordo com a temperatura, além da subida do R3. +frio -> +tensão

▲ Rise time?
 ADP1755
 MAX8516



FONTES FIXAS P/ PCU JÁ QUE SEUS COMPONENTES TOLERAM GRANDES FAIXAS DE TENSÃO

LP38693MPX-ADJ (idiv > 100uA, R2<12K)

$V_{OUT} = 1.25 \times (1 + (R1 / R2))$
 $R1 = 20.5KOHMS$
 $R2 = 10KOHMS$

$V_{OUT} = 3.8125V$

MAX8880 (R4 = 1.2M, idiv > 1uA)

$V_{OUT} = V_{FB} \times (1 + (R1 / R2))$
 $V_{FB} = 1.257V$
 $R1 = 2.44MOHMS$
 $R2 = 1.2MOHMS$

$V_{OUT} = 3.8129V$

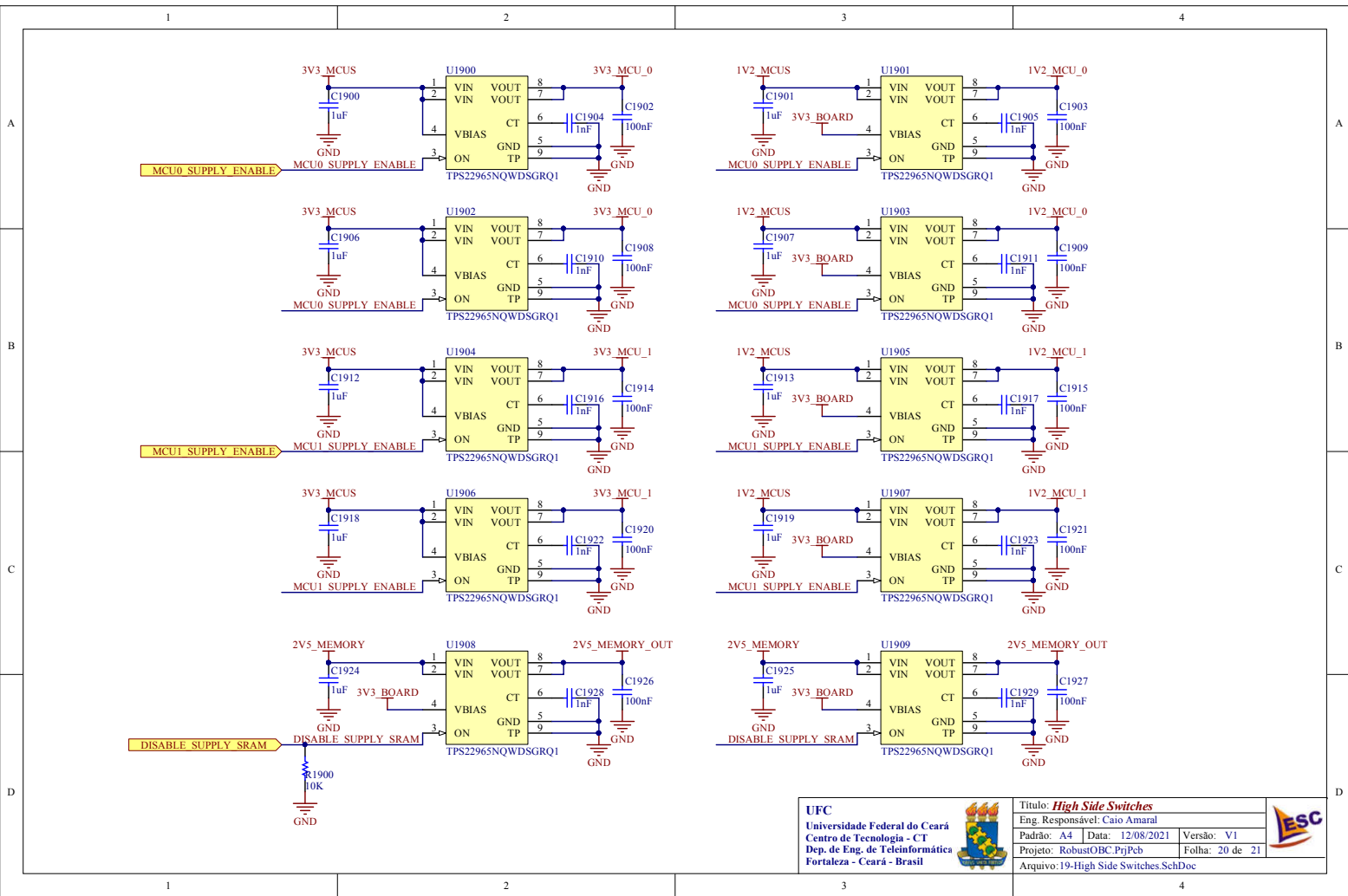
NCV8537 (10uA > idiv > 100uA)

$V_{OUT} = 1.25 \times (1 + (R1 / R2))$
 $R1 = 73.2KOHMS$
 $R2 = 35.7KOHMS$

$V_{OUT} = 3.813V$

VCC3V3_PCU VAI VARIAR NA FAIXA DE 3.71V ATÉ 3.5V CONTANDO COM A QUEDA DO DIODO



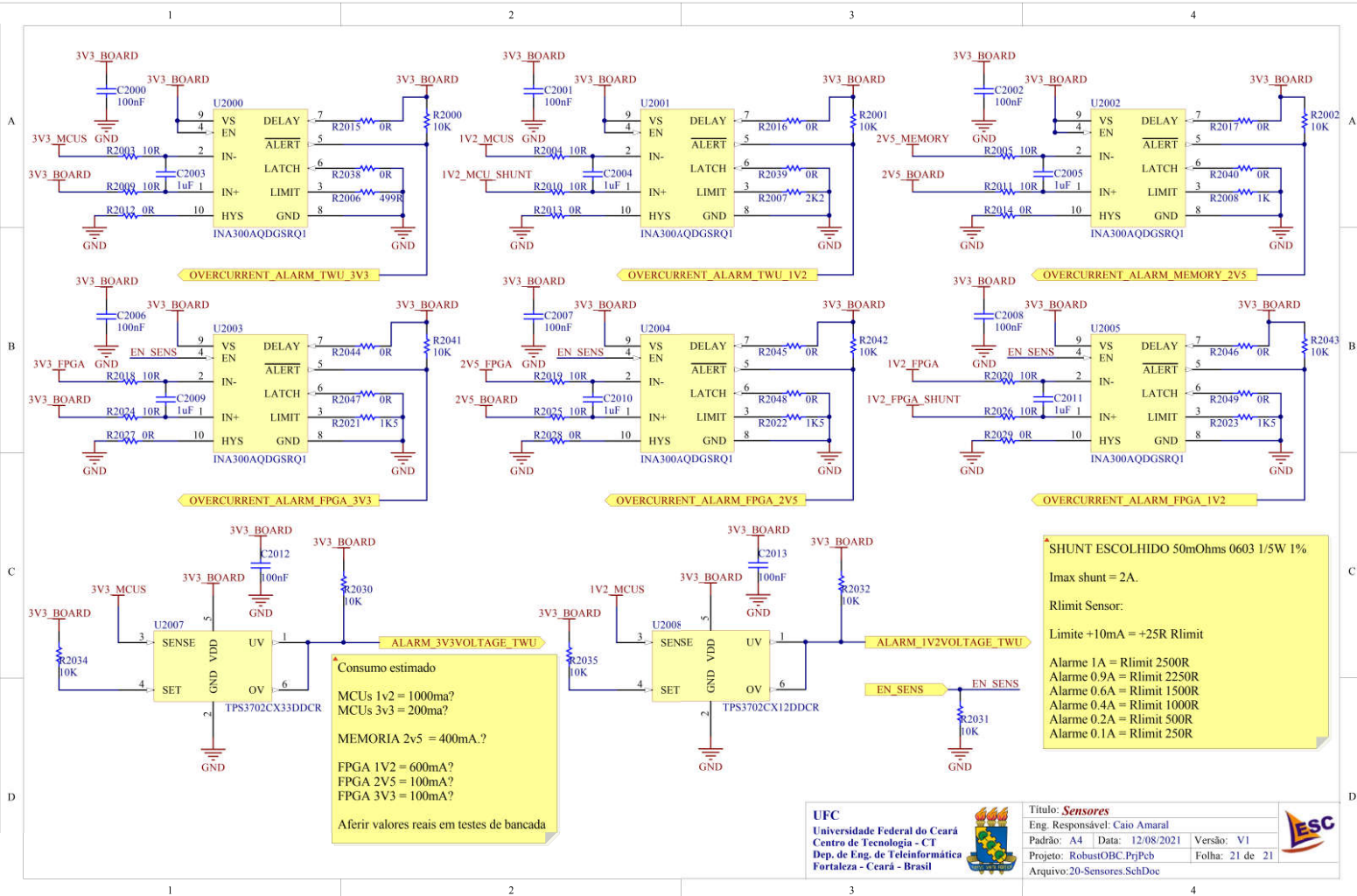


UFC
 Universidade Federal do Ceará
 Centro de Tecnologia - CT
 Dep. de Eng. de Teleinformática
 Fortaleza - Ceará - Brasil



Título: High Side Switches		
Eng. Responsável: Caio Amaral		
Padrão: A4	Data: 12/08/2021	Versão: V1
Projeto: RobustOBC.PrjPcb		Folha: 20 de 21
Arquivo: 19-High Side Switches.SchDoc		





Consumo estimado
 MCUs 1v2 = 1000mA?
 MCUs 3v3 = 200mA?
 MEMORIA 2v5 = 400mA?
 FPGA 1V2 = 600mA?
 FPGA 2V5 = 100mA?
 FPGA 3V3 = 100mA?
 Aferir valores reais em testes de bancada

SHUNT ESCOLHIDO 50mOhms 0603 1/5W 1%
 Imax shunt = 2A.
 Rlimit Sensor:
 Limite +10mA = +25R Rlimit
 Alarme 1A = Rlimit 2500R
 Alarme 0.9A = Rlimit 2250R
 Alarme 0.6A = Rlimit 1500R
 Alarme 0.4A = Rlimit 1000R
 Alarme 0.2A = Rlimit 500R
 Alarme 0.1A = Rlimit 250R